ParallelGraphics

Internet Scene Assembler 2.0

User's Guide

Copyright © 1999-2002 ParallelGraphics

Contents

INTRODUCTION 1
What is Internet Scene Assembler? 1
About this Manual 1
Document Conventions 1
Internet Scene Assembler Features 1
What's New in ISA 2.0? 2
Removing ISA 4
Getting Help 4
Getting Assistance 4
Technical Support 5
ParallelGraphics Web Site 5
GETTING STARTED 6
Screen Layout 6
Working with ISA 8
Customizing Toolbars 9
Keyboard Shortcuts 9
Control Gallery Resources 10
Quick Tour 11
Lesson 1. Assembling Objects and Adding Lights 11
Lesson 2. Creating a New Animated Object 14
Lesson 3. Editing an Object 16
ASSEMBLING A SCENE 19
Creating, Opening, and Saving 3D Scenes 19
Create a New Scene 19
Use Background Schemes 19
Open an Existing Scene 20
Import a Scene 20
Save a Scene 22
Save Selection 22
Prepare a Scene for Publication 22
Setting Publishing Options 23
Publishing Warnings 24
Undo/Redo Operations 24
3D Navigation 25
Move a Camera in the Perspective View Window 25
Using Viewpoints 25
Moving Around: Walk, Fly and Examine 26
Restore, Fit and Align 29
Adjusting a Camera 29
Keyboard Commands 29
Move a Camera in the Plan Window 30
Setting Perspective Window Options 31
Using Objects 33
Overview of Using Objects 33
The Scene Tree 34
Place Objects into a Scene 34
Select Objects 36
Delete Objects 37
Rename Objects 38
Drop Objects 38
Align an Object 38
Move Objects to Another Hierarchic Level 38
Cut, Copy, and Paste Objects 39
Replace an Object 40

Save Object ... 40
Save Scene as Object ... 41
Edit Object ... 41
Insert Objects from Other Scene ... 42
Find an Object ... 42
Freeze an Object ... 42
Wrap by Group ... 43
Using Transforms ... 43
Placement ... 43
Manipulate objects in the Plan window ... 44
Using the Transform Manipulators ... 44
Moving and Rotating Objects ... 45
Scaling Objects ... 45
Choosing a Transform Center ... 46
Correcting a Transform Center ... 46
Using the Transform Type-In ... 47
The Axis Constraint Buttons ... 47
Manipulate objects in the Perspective View window ... 47
Object Manipulator Controls in the Perspective View Window ... 48
Rotate an Object Directly in the Perspective View Window ... 48
Scale an Object Directly in the Perspective View Window ... 49
Move an Object Directly in the Perspective View Window ... 49
Replacing with the Object from the Object Gallery ... 50
Manipulating Plan ... 50
Choosing a Reference Coordinate System ... 50
Global coordinate system ... 50
Local coordinate system ... 51
Screen coordinate system ... 51
Change a View ... 51
Move a Plan (Pan the View) ... 52
Rotating a Plan ... 53
Zooming a Plan ... 53
Fit the Plan to an Object ... 54
Fit Plan to a Scene ... 54
Using the Grid ... 54
Guide Lines ... 55
Turn the Follow Selection On/Off ... 55
Keyboard Commands ... 56
Setting Plan Window Options ... 56
MAKING A SCENE INTERACTIVE ... 58
Overview of Fields, Events, and Routes ... 58
Operating with the Properties of Objects ... 58
Objects' Properties ... 58
Viewing Properties ... 59
Change Properties ... 59
Properties Editing Interfaces ... 60
texture ... 60
material ... 61
Designing Materials ... 63
SFString and MFString ... 63
Object Interaction ... 64
Interaction of Objects ... 64
Events ... 64
Routing in the Properties/Events Window ... 65
Routing in the Routing Diagram Window ... 66
Rules for Routing ... 67
Forwarding Events to Console Window ... 67

Manipulating Objects in Routing Diagram 67
Move an Object in the Routing Diagram 67
Hide or Show Objects 68
Find the Selected object 68
Turn the Snap Valid Routes On/Off 68
Colors Preferences Settings 69
Advanced View 69
Animating Objects 69
Animation 69
Animator Toolbar 70
Create an Animation 71
Preview an Animation 72
Controlling Animation 72
Selecting and Deselecting a Keyframe 73
Delete a Keyframe 73
Cutting, Copying and Pasting Keyframes 73
Animating a Viewpoint 74
Debugging 75
Console Window 75
CREATING AN ISA OBJECT 77
Object Interface Control 78
Editing existing ISA object 80
Creating Internet Scene Assembler Objects Using Internet Space Builder 81
USING AVI RECORDER 82
Key Features of AVI Recorder 82
Record a Movie 82
AVI Recorder Options 82
PROPERTIES REFERENCE OF OBJECTS 84
Touchable – TouchSensor 84
Hidable – Cloak 84
Collidable – Collider 85
Movable – Mover 85
Turnable – Rotator 86
Proximity – ProximitySensor 87
Looksensitive – VisibilitySensor 88
Billboarded – Billboard 88
Highlighted – DirLight 89
Labeled – Label 89
Linked – Anchor 90
Camera-aligned – Pin 90
DropObjectTarget – DropObjectSensor 91
DropTextureTarget – DropTextureSensor 91
APPENDIXES 93
Troubleshooting 93
ISA Limits 93
Object Gallery 93
CHARACTERS 93
A Walking Character 94
PathFinder 94
CheckPoint 95
Example with Characters 95
CONTROL 96
Delay 96
DelayBool 97
DemoControl 97
EventCounter 98
Eye_to 99

JSfunction 99
Menu 100
Slider 101
StepControl 101
TimerToggle 102
IMAGE 103
Movie 103
Picture 103
TextureMover 104
TextureSlideShow 104
LOGICAL 105
AdapterBool 105
AdapterTime 105
Boolsorter 106
Boolean Filter 106
Time Filter 106
Inverter 107
Logical AND 107
Logical OR 107
Logical XOR 108
Selector 109
Selector Bool 109
SwitchBool 110
SwitchNode 110
SwitchTime 111
Time Bool Converter 111
WorldEntry 112
WorldEntryBool 112
NAVIGATOR 112
Live Camera 112
Map 113
Locator 114
SYSTEM 115
Background 115
Triggered Button 116
DirLight 116
Fog 117
Group 117
Inline 117
Link 118
LOD 118
Navigation Info 119
PointLight 119
Proximity Sensor 120
Sound 120
SpotLight 121
Timer 122
Viewpoint 122
Visibility Sensor 123
WorldInfo 124
TEXT 124
GetString (the string generator) 124
toStrConv (the string converter) 125
CheckBox 125
RadioButton 126
RadioButtonControl 127
Text 127

Text3D 128
TRANSFORM 129
MoveRepeater 129
RotationInterpolator 129
RotationRepeater 130
Screw 131

Introduction

What is Internet Scene Assembler?

Welcome to Internet Scene Assembler Pro (*ISA*) – the 3D authoring tool from ParallelGraphics. *ISA* facilitates the creation of interactive and dynamic 3D scenes that involve complicated logic and behavior such as remote training manuals, maintenance and support guides and interactive product presentations amongst others.

3D scenes created with *ISA* can be published on the Web or integrated into a stand-alone application written in C/C++, Visual Basic, Delphi, or any third party application supporting ActiveX (like MS PowerPoint), or a database.

About this Manual

This manual is divided into seven chapters and three appendixes. The first two chapters provide an introduction to *ISA,* installation instructions and contain tutorials and examples designed to teach you how to use the major features and objects of *ISA*. The appendixes cover tips and tricks, *ISA* limits and objects included in *ISA* galleries.

Document Conventions

Conventions used in this manual have been established to help you learn to use *ISA* quickly and easily. As much as possible, the conventions correspond to those found in other Windows documentation.

Each section contains an overview discussion, typically preceded by a path annotation showing how to access the feature in the program. Italic type is used for path annotations, file and field names. Dialog box, gallery and VRML node names begin with uppercase letters. For example, the Open dialog box, the System gallery. Boldface type is used to emphasize the focus of the discussion.

Key combinations are indicated by the name of a modifier key and an action. For combination “SHIFT+F1”, you press and hold the first key while you press the second key. “SHIFT-drag” means “hold down the SHIFT key, hold down the mouse button and move the mouse.”

Internet Scene Assembler Features

ISA includes features that help you work conveniently and quickly – including the following:

- Assembling 3D scenes from static and animated objects;
- Animating objects;
- Creating customized galleries of reusable 3D objects (this allows you to apply a module-based approach in creating 3D scenes);
- Establishing relations between events to create complex objects' behavior;
- Adding interaction between users and scenes;
- Creating guided tours through 3D scenes;
- Module-based approach in creating 3D scenes.

ISA offers you visual support for all operations with objects which are clearly displayed in specialized windows:

- 3D/2D view of objects and scenes (the Perspective View or Plan windows);
- The animation parameters for objects (the Animator window);
- The scheme of relationships between events (the Routing Diagram window);
- Object properties and events (the Properties/Events windows);
- Object hierarchy (the Scene Tree window).

ISA allows you to import and export scenes and objects in VRML97 format. You can also import VRML 1.0 objects and scenes in *ISA* by using Cortona VRML 1.0 Converter which can be downloaded from the ParallelGraphics Web site at: http://www.parallelgraphics.com/products/cortona/converter/

The following 3D VRML objects can be used in *ISA* :

- CAD models converted into VRML97 format. It is recommended that you use ParallelGraphics Internet Model Optimizer (http://www.parallelgraphics.com/products/imo) to simplify CAD 3D models reduce their file size.
- Objects created with the help of 3D modeling software tools, (for instance, 3D Studio Max), which export their 3D models in the VRML97 file format.
- Objects created with the help of specialized VRML authoring tools, including ParallelGraphics Internet Space Builder, VrmlPad and Extrusion Editor. For more details, see http://www.parallelgraphics.com/products/

What's New in ISA 2.0?

Plan window

- View the scene using new orthographic projections and define your own projection if necessary.
- Switch between the coordinate systems for object manipulation.
- Switch between the object manipulation modes and the plan manipulation modes.

Routing Diagram

- Reconnect and copy the routes.
- Color the object border and outgoing routes to make the routing diagram easier to follow.
- Use the new Advanced View mode for routing the transform fields of objects.
- Mark the outgoing events for debugging.

New debugging facility

- The new Console window allows you to debug the scene in the Preview mode.
- You can track the events marked in the Routing Diagram window.

Operating with Object Properties

- Use Material Editor to change the material exposed property of an object and animate it.
- Change and animate the textureTransform exposed property of the object.

Operating with Object Abilities

- Some new Abilities have been added to the Properties tab.
- The Turnable and Movable abilities have also been improved.

Key Frame Animator

- Copy, paste, and reverse the animation keys.

Scene Tree window

- The improved Scene Tree window allows you to arrange objects in an easier way.

Galleries

- Use the new "smart" objects from the object galleries to make your scenes interactive:

  CONTROL – developing 3D demonstrations and manuals.
  IMAGE – applying and modifying pictures or textures.
  LOGICAL – controlling the events.
  NAVIGATOR – maps and a live camera.
  TEXT – text menus, radio-button and check box controls.
  TRANSFORM – simulating commonly used animations.
- For those objects with the texture field exposed in the object interface, you can change the textures by dragging and dropping them from the Texture Gallery.

Import command

- *ISA* now allows you to import interactive and animated scenes that contain some behavior logic.
- The interactive capabilities of a scene are retained.
- The Viewpoint nodes used in the scene are converted to *ISA* objects and are accessible in *ISA*.

Publishing

- Now you can optimize your scene for publication on the Web. All "auxiliary" information generated and used by *ISA* can be excluded from the source code of the scene.

Creating and editing ISA objects

The Save Scene as Object command allows you to create and edit the object interface more easily.

- The names of exposed properties are now generated automatically.
- The material and texture Transform properties can now be exposed.
- The Exposed Field/Events and the Scene Tree windows in the Save Scene as Object dialog are now synchronized.
- You can now publish the created object.
- The Edit Object command allows you to edit the interface of any object selected in the 3D scene.

Assembling a Scene

- The Find Object command from the Edit menu allows you to easily find the object in a complicated scene.
- The improved Replace Object command from the Object menu allows you to easily replace any object used in the scene.
- The Save Selection command from the File menu allows you to save a part of the scene in a separate file to create reusable templates.

Removing ISA

To remove the *ISA* program:

1. Click the Start button, and then point to Settings.
2. Click the Control Panel, and then double click the Add/Remove Programs icon.
3. In the dialog box select Internet Scene Assembler and click the Change/Remove button.
4. Click Next if you want to remove the ISA installation completely. Click Cancel if you decided to keep ISA.
5. It is strongly recommended that you click the Yes to All button in the Remove Shared Component dialog.

Notes:

- The Wise uninstaller program deletes all the standard *ISA* components, standard gallery files and registry entries. You must manually remove any shortcuts to *ISA* that you may have created elsewhere.
- The Wise uninstaller program does not delete the elements you've created by yourself in any gallery. The corresponding gallery folders are also not deleted. You may use them later with the newly installed *ISA*.

Getting Help

Getting Assistance

You can get help without interrupting your work.

For a brief description of a menu command:

- Position the pointer over the command. The description appears on the Status bar – or
- Position the pointer over the menu item and press F1.

For a description of a toolbar button:

- Position the pointer over the button. The name appears below the button – or
- Press SHIFT+F1 and then click the button.

Objects from the object gallery may have short explanations (see Appendixes, Object Gallery for details) for the object's events and properties. If the object is included into the scene, select the object, and click the Help tab in the Properties/Events window to see the explanation.

Technical Support

We are interested in receiving your comments and suggestions about *ISA*. For product support, send an e-mail to support@parallelgraphics.com. Before posting your message, please follow these simple instructions:

1. Check the online Help.

2. Include in your message any information that could help answer your question or solve the problem.

3. Document the system configuration: OS version, CPU type, memory, sound card, graphics card, monitor, etc.

4. When error messages occur please document the complete message including sub-dialog details and provide step-by-step instructions for reproducing the problem.

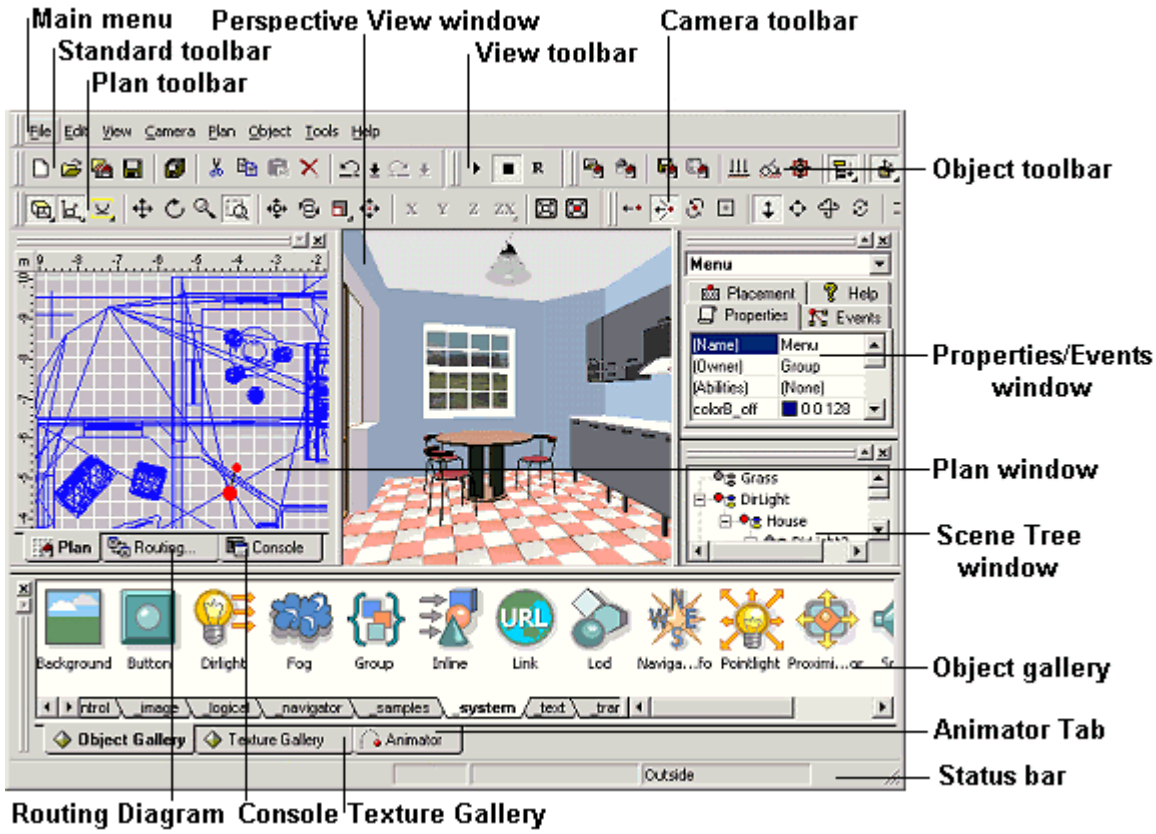
ParallelGraphics Web Site

Visit http://www.parallelgraphics.com on the World Wide Web for the latest information on our company, future product releases, and as a central location for visiting other 3D spaces.

Getting Started

Screen Layout

When you start *ISA* you see an arrangement of screen elements, which is described below.



Plan window

Different views of the scene. By default, the top view of 3D scene is shown. You can change the view using the flyout menu items on the Plan toolbar.

Perspective View window

3D scenes are viewed here. You may navigate the camera using the mouse or keyboard. ParallelGraphics Cortona VRML client is used as a viewer in the Perspective View window.

Properties/Events window

This allows you to display and set the object's properties and events connection.

Scene Tree Window

This window is used for viewing and selecting objects within the hierarchical structure of the current scene.

Routing Diagram tab

Allows you to switch to the Routing Diagram window.

Console tab

Opens the Console window which allows you to view events during a scene preview.

Object gallery

Objects that can be added to your scene. Objects are grouped by category and are displayed by clicking a category's tab at the bottom of the gallery window.

Texture Gallery tab

Opens the Texture gallery. This is a collection of textures sorted by categories. For the objects with the *texture* field exposed in the object interface, you can change the textures by dragging and dropping them from the Texture gallery.

Animator window tab

Opens the Animator window. It is used to animate the object in the scene.

Main menu bar

This is located at the top of the screen. Some menu commands duplicate the buttons in the toolbars.

Standard toolbar

Buttons for managing files, help and window control.

Plan toolbar

Controls for Plan window modes, zooming, fitting plan to the window, and dropping objects on the underlying face.

View toolbar

Buttons for controlling the viewing mode of *ISA* and recording a movie.

Object toolbar

Object operations.

Camera toolbar

Controls for navigation modes, collider, and headlight.

Status bar

Located at the bottom of the *ISA* screen. Here you can read short explanations for commands and program status information.

ISA's title bar (not shown on the figure)

Located at the top of the *ISA* screen. You can open an existing scene by dragging the file onto it.

Flyouts

There are also flyouts that may have additional commands in the sub-menu. A flyout is an icon-based menu available from any button that has a small black triangle in the lower-right corner, for example, . To access the flyout, click and hold the button. Then to activate a flyout menu item, drag to the item and release.

Working with ISA

This section provides suggestions for ways you can use *ISA* to create a virtual 3D space and add behavior to objects.

- Create a new scene. Import or open an existing scene.
- Add objects to your scene from a gallery of pre-built objects. These objects can be positioned and resized. The Object gallery also includes special groups of Characters, Control, Image, Logical, Navigator, System, Text and Transform objects.
- Assign to the objects you want the additional features – abilities.
- Make animations for appropriated objects in the Animator.
- Create routes between objects through output and input events.
- To make sure your scene looks correct within a VRML browser, choose Play from the View menu or click ▸ Play on the View toolbar. This enters the Preview mode of *ISA*. You can explore your scene using the navigating commands in the Perspective View window and to check how objects act and react. You cannot modify a scene during its preview. In order to make a modification, such as adding or removing an object, or creating animation or changing an object's properties or routing, make sure the scene is not previewed.
- You can exit Preview by clicking ■ Stop on the View toolbar any time you need to. This will enter the Design mode and allow you to then make corrections.
- The Save Scene as Object command allows you to create and edit a new object for a gallery.
- Save and publish the scene.

When working with *ISA*, you can give commands in several ways:

- Menu items.
- Toolbar buttons.
- Keyboard shortcuts.

- Context dependent pop-up menus (click the right mouse button to display).
- Drag and drop operations.

Customizing Toolbars

To show or hide a toolbar:

- Choose Toolbars from the View menu and select the bar you wish to show or hide.

To move a toolbar:

1. Position the pointer over a blank area on the toolbar. Do not point at the one of the buttons.
2. Press and hold the left mouse button.
3. Drag the bar by moving the mouse to a new location. If you drag it to the edge of an *ISA* window, the toolbar docks.
4. Release the mouse button.

To show or hide a toolbar button:

1. Choose Customize Toolbars from the Tools menu. The Customize Toolbars dialog box appears.
2. In the box to the left click the category that contains the desired button. All buttons of this category appear in the Buttons box to the right.
3. Mark or clear the check box for the button you wish. Click OK.

To make an anchored toolbar a floating toolbar – or vice versa:

- Double-click a blank area on the toolbar.

Keyboard Shortcuts

Tools menu > Keyboard Shortcuts > Shortcut Keys dialog box.

You can assign a shortcut key to a command:

1. On the **Tools** menu, click **Keyboard Shortcuts**.
2. In the **Select a macro** box, click the command you want. In the **Assigned shortcuts** *ISA* displays any shortcut keys that are currently assigned.
3. Click the Create Shortcut button and then press the new shortcut key you want to assign. Click OK.

To restore shortcut key assignments to their original settings:

- In the **Shortcut Keys** dialog box, click the **Reset All** button. Click OK in the *ISA* confirmation dialog. This removes all changes that have been made previously.

Control Gallery Resources

Right-click anywhere in the gallery. The pop-up window appears.

Refresh

Rereads the gallery contents.

Tool Tips

Disables or enables previewing image tooltips.

Titles

Disables or enables previewing image titles.

Properties

Shows the properties of the current gallery.

Path to the resources. Enter the name of the folder with the gallery resources or click the Browse button to search for the gallery folder.

Include subfolders. Mark the check box to use resources from the sub-folders of the folder specified in the Path to the resources edit box. If the check box is marked the sub-folders are shown in the corresponding gallery with a tab. The name of the tab coincides with the name of the sub-folder.

Preview size. Specify the size of the screen area for the resource preview. The size is specified in pixels.

Sort by. Define the sort order for the resources. They can be sorted in ascending order by name, or in ascending or descending order by the creation date.

Show ToolTips. Mark the check box to display tool tips for gallery resources.

Show titles. Mark the check box to display names of gallery resources.

Allow fast load. Mark the check box to save a resource preview image after the resource is loaded. This option is intended for complex resources such as scenes or objects. These resources are displayed much faster when their preview images are stored on the disk. If you modify a resource its image is automatically updated.

Quick Tour

This is step-by step tutorial in which you build several scenes and learn to perform basic operations in *ISA*. The lessons in the tutorial use files provided with the *ISA* program. The interim files are installed as part of the default installation in the *Program Files\ParallelGraphics\ISA\Samples* folder.

Lesson 1. Assembling Objects and Adding Lights

This lesson demonstrates how to set up the background scene, add objects, understand a scene hierarchy, perform simple manipulations in a plan, add and specify an object's properties such as the TouchSensor trigger to turn the light on, create routes and publish the finished scene. In this exercise, you'll create a simple scene containing the room, a table and the lamp that can be lit as a result of the user click. The *Sample1_1.wrl* sample scene, which is available in the *"…\ISA\Samples \Sample1"* folder uses the following objects:

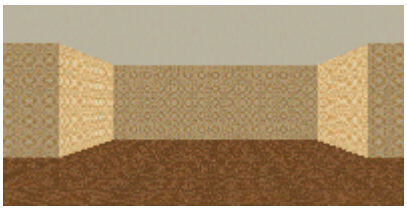
DirLight, room, Viewpoint, PointLight, Navigation_Info, table_comp, and lamp.

Step 1. Specifying the background scene and opening a new file

1. Start *ISA* by clicking **Start**, point to **Programs**, point to **ParallelGraphics** and then click **Internet Scene Assembler**. Maximize the program window. The newly created scene does not have any background image or landscape, but *ISA* gives you several pre-built background schemes for your virtual world.

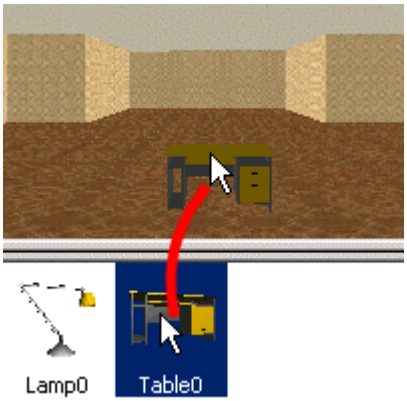
2. On the **Tools** menu point to **Options** and click **Set Environment**. Select the **scene_room.wrl** scheme and then click **Open**. The room appears in the scene whenever you create a new scene.

3. Create a new scene by clicking the New button on the Standard toolbar. A part of house with decorated walls appears in the Perspective View window:



Step 2. Adding objects

1. Click the **Samples** tab in the Object Gallery, position the pointer over the Table0 thumbnail, press the left mouse button and move the mouse while holding down its left button to position the pointer on the floor in the Perspective View window. Release the left button. The table appears in the scene in the place where you have dropped it:



Please note that the object appears selected. Therefore the object rotation controls (green cones at the end of the axes crossing the object center) and object resizing controls (yellow boxes in each bounding box vertex) are also visible. To turn off the selection, click anywhere on an empty part of the Scene Tree window. To prevent occasional movement of the room, you can declare this object as "frozen": right-click the **room** object in the Scene Tree window and choose **Freeze room**. Frozen objects remain in the scene, but you can't select, transform or modify them. The icons of frozen objects in the Scene Tree window turn gray. Also freeze **DirLight** and **DirLight(2)**. These light sources illuminate the room and are parent objects to the room.

2. Drag the **lamp0** object on the table in the Perspective View window. Please note that the lamp becomes a child of the table because it was dropped on the table. This hierarchy linkage of objects is represented in the Scene Tree window:

3. From the System gallery, drag the **PointLight** object on the lampshade where the bulb should be.

Step 3. Moving objects in a plan

1. *ISA* provides the Plan window, on the left by default, that enables control over the positioning and alignment of objects in 3D space. To place the PointLight object exactly in the lampshade, set up the Global reference coordinate system for manipulations in a plan. On the **Plan > Coordinate System** menu, choose **Global**. The Global coordinate system is related to the VRML world as a whole and specifies the coordinate system used for a transformation (Move, Rotate, Scale).

2. By default the Plan window shows the top view. Make sure the Plan Projection is set to Top: **Plan > Projection > Top**.

3. To make the PointLight object visible in the plan, select it by clicking **PointLight** in the Scene Tree window, and then use the Fit to Object command: **Plan > Fit to Object**. The PointLight object appears on the plan in a wireframe (a red sphere). To increase or decrease the view magnification, click the **Zoom Plan** button on the Plan toolbar, position the pointer in the center of the sphere and press the left mouse button. The position where you have pressed the button determines the point on the plan about which zooming will take place. Move the mouse up or down to increase or decrease the view magnification. Release the left mouse button.

4. Specify a manipulation mode in the plan: **Plan > Manipulation Mode > Move Object**.

5. Constrain transformation to the ZX-plane: **Plan > Restrict to > ZX Plane**. This allows you to move the selected object in the horizontal plane.

6. Position the pointer on any line of the sphere or over its pivot point. The pointer changes to . You can now move the point light to place it in the center of the lampshade by dragging its sphere in the Plan window.

Step 4. Specifying the light emission

1. Make sure the PointLight object is selected. Click the **Properties** tab in the Properties/Events window.

2. Click anywhere in the **color** row to activate the corresponding property of the PointLight object. Click the appeared button and select the color from the drop-down palette. Try different colors. Finally, we suggest you to simply type **255 255 145** and press ENTER to apply the color.

3. Set the **radius** property to **10** m. In this case the walls of the room should not be lit.

4. Set the **SphereRadius** property to **0.1** m. This defines the radius of the sphere that represents the PointLight object in the Plan window. The smaller sphere better suits to the current lamp. Note: you cannot scale the PointLight object using the object manipulation controls since this changes the radius property accordingly.

5. Set the **On** property to **False**. The light is initially turned off.

Step 5. Saving a scene

1. On the **File** menu, choose **Save**. A dialog box for the file name appears. In the File name box, type a name for the VRML scene. Make sure the VRML Files item in the Save as type box is selected to save the scene in the VRML 97 format. Click Save.

2. Note that the already created scene of the current step is available in *"…\ISA\Samples \Sample1\Sample1_1.wrl"*.

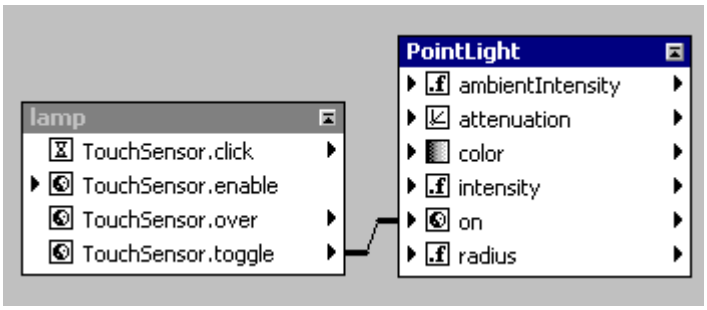
Step 6. Making a lamp interactive

1. Select the **lamp** object.

2. Adding a Touch Sensor. This allows you to detect the mouse click over the lamp and generate events capable of changing the object's properties. In the Properties/Events window at **Properties** tab, click **Abilities**, then the appeared button and select **Touchable** from the drop-down dialog box. As you position the pointer over the object with the Touchable ability assigned, the pointer changes to in the Preview mode or Cortona VRML client.

3. Click the Routing Diagram tab. The Routing Diagram window appears instead of Plan window. Hide all objects except **lamp** and **PointLight**. To do that, right-click anywhere outside the object's icon in the Routing Diagram window and choose Hide/Show Objects. Clear the corresponding check boxes in the Hide/Show Diagram Object dialog box then click OK.

4. Create a route. Position the pointer over the right-pointing arrow located at the right of the **TouchSensor.toggle** of the lamp object (outgoing event). Press the left mouse button and holding it down, stretch the route to the right-pointing arrow, located at the left of the **on** of the PointLight object (incoming event):



Step 7. Previewing a scene

1. Enter the Preview mode by clicking the ▶ Play button on the View toolbar.

2. Examine the lamp by clicking it. The light turns on and off.

3. Exit the Preview mode (this enters the Design mode) by clicking the ■ Stop button on the View toolbar.

Step 8. Saving a scene

1. On the **File** menu, choose **Save**. This saves your scene for further editing.

2. Note that the already created scene of the current step is available in *"…\ISA\Samples \Sample1\Sample1.wrl".*

Step 9. Publishing

The publishing procedure prepares a scene for uploading to the Web site or for use on another computer. *ISA* creates a new folder in which all data describing the scene is placed. This sub-folder will contain the scene's *.wrl* file and other resource files, for example, textures, sounds, pictures, etc. By setting the publishing options (Tools menu > Options > Publish) you can control the internal structure of the VRML scene, select a folder where all scene resources should be stored, and significantly reduce the physical size of the final VRML (WRL) file with optimization.

- On the **File** menu, choose **Publish**. Specify the publishing options by clicking the Options button in the appeared dialog box if you want to change the default settings.

Note:

- The optimization process removes *ISA* specific information making it impossible to further edit a scene.

- The already created scene of the current step is available in *"…\ISA\Samples \Sample1\ Sample1_pub.wrl".*

Lesson 2. Creating a New Animated Object

The aim of this lesson is to acquaint you basis animation technique and the interface with an object. You learn the following: how to import the VRML model; create a simple animation; create a new *ISA* object with the Save Scene as Object command; and edit routes. In this exercise you'll create a button that stays put with the first click and resets with the second.

Step 1. Importing a readymade VRML model

- We'll use the readymade model of the button which is available in *"…\ISA Samples\Sample2\Sample2.wrl"*. You can easily build a similar model by using one of the common available VRML modelers, for example, ParallelGraphics ISB 3.0, Extrusion Editor V.1.0 for VrmlPad. Import the model of the button by using the Import command: choose **File menu > Import**, navigate to the *ISA\Samples\Sample2* and double-click **ButtonGeometry.wrl**. The button consists of two elements linked together: BUTTON_base as a parent object and BUTTON as a child object. Note that the name in the Scene Tree window corresponds to the name under the DEF statement, followed by the Transform node in the VRML file of the imported model. The scene of this current step is available in *"…\ISA\Samples\Sample2\ Sample2_1.wrl"*.

Step 2. Animation

1. Select the BUTTON object.

2. Create a new animation for the selected object: click the **Animator** tab and click the **New Animation** button on the Animator window toolbar.

3. Make the button visible in the Plan window from the front view: **Plan > Projection > Front**. Choose the local reference system to move the object in the coordinates of its parent: **Plan > Coordinate System > Local**. Now you can move the BUTTON by dragging its Y-axis in plan.

4. Record keyframes: move the selected BUTTON object to the new position of locked button, drag the time slider (the yellow diamond) to the time 0.5, click the Record All Changes button on the Animator toolbar. Note that the first keyframe at 0 is recorded automatically.

5. Add the flag that stops the animation at 0.5. To do that, click **Toggle the Stop Milestone** on the Animator toolbar.

6. Copy the first keyframe to 1.0. To do that, drag the first blue dot while holding down the CTRL key to 1.0.

7. Add the flag that stops the animation at 1.0. To do that, drag the time slider to 1.0 and click **Toggle the Stop Milestone**.

8. Click **Play** to preview the animation then click **Stop** to stop paying and click **Rewind** to restore the initial button position.

9. Define events that trigger the animation when the user clicks the button: click **Trigger** on the Animator toolbar and choose **TouchSensor**. See “Controlling Animation” for more details.

10. Examine the animation in the Preview mode: **View > Play**, click the button in the Perspective View window to test. Use **View > Stop** to enter Design mode when you finish. The scene of this current step is available in *"…\ISA\Samples \Sample2\Sample2_2.wrl"*.

Step 3. Using the Save Scene as Object command

One of the most common ways of using the Save Scene as Object command is to create animated objects with the interface specified for the *ISA* object gallery.

1. On the **Object** menu, choose **Save Scene as Object**. The Save Scene As ISA Object dialog appears.

2. Under the **TouchSensor** group in the "Object definition" box, expose the following fields end events by double-clicking them: **Enabled**, **click**, **toggle**, **over**, **enable**.

3. Shorten the names of fields end events in the Object interface list box by removing the name prefixes: right-click the item and choose **Rename**. The new names are as follows: **Enabled**, **click**, **toggle**, **over**, **enable**.

4. Specify the object name: in the **Object name** box, type **Button1**.

5. Save the object in the Object gallery: Click **OK**, and then **Save** in the Save As Object dialog. The default folder name is Objects and the Button1 object appears in the Object gallery under the General tab. You can also create a new folder directly in the Save As Object dialog by clicking the Create New Folder button.

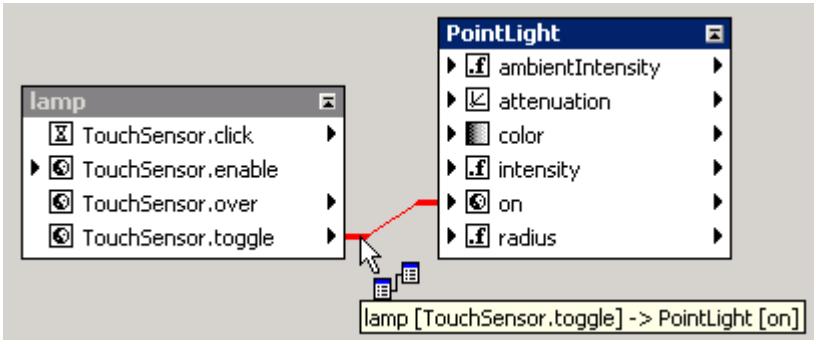
6. The readymade object for this step can be found in Samples gallery.

Step 4. Applying the Button object

1. Open the scene from the previous Lesson 1: *"…\ISA\Samples \Sample1\ Sample1_pub.wrl"*.

2. Add the Button object to scene from the gallery where you’ve saved it or the Button object from the Samples gallery. Place the button anywhere on the table close to the lamp.

3. In the Routing diagram window, reconnect the start of a route in such a way that it begins from **Button[toggle]**: position the pointer over the route joint you want to move,



press the left mouse button and holding it down move the route to **Button[toggle]**. The *toggle* outgoing event of the Button object generates the Boolean value TRUE when the user clicks the button to press it and FALSE when released.

4. Remove useless TouchSensor ability from the lamp object. To do that, select the lamp, click **TouchSensor** at the Properties tab of the Properties/Events window and the button which appears.

5. Save the scene. The readymade scene for this current step can be found in *"…\ISA\Samples \Sample2\ Sample2.wrl"*.

Lesson 3. Editing an Object

You can customize the materials in the material palette to create the effect you want by editing their properties in the Material Editor. As you see when you open the Material Editor, materials contain many other characteristics besides color that effect their appearance on the screen. In this lesson, you change the object interface to expose the *material* field, and then play with the color. In addition, you learn how to replace the object in the scene.

Step 1. Opening the object for editing

1. The model of the button is available in *"…\ISA\Objects_Samples\Button1.wrl"*. You can open it by using the Open command from the File menu. You can also open an existing model by dragging its thumbnail from the Samples gallery onto *ISA's* title bar. See " Open an Existing Scene" for details on using these commands.

2. To make the button fully visible in the Perspective View window, use Go To Object on the Object toolbar (see "Adjusting a Camera").

Step 2. Exposing material fields

1. Select the **BUTTON_base** in the Scene Tree window. Click **Object > Edit Object**. This opens the Edit Object dialog in which you can specify a new object interface.

   In the "Object definition" box, click the square icon with the plus next to the **appearance [Exp]** field to expand the hierarchy level. Double-click the **material [Exp]** field to add it to the object's interface. The exposed field appears in the interface list on the lower-left. The red icon in the "Object definition" box represents the field or event selected in the Object interface list. Click **OK**. Now the material field becomes accessible in the Properties/Events window at the Properties tab and you can specify the material field by using the Material Editor.

2. To start the Material Editor, click the material field at the Properties tab, and then the button that appears. Now you can see all the materials that have been already designed. If the Auto Apply check box is selected, clicking the material sample in the Sample Collection results in changing the material of the selected object in the scene. Try different materials, and then experiment with other options. Click OK when you finish. All changes can be undone

3. Select the **BUTTON** in the Scene Tree window and expose the material field by repeating the item 2.

Step 3. Saving the object

1. Click **Object > Save Scene as Object**. This opens the Save Scene As ISA Object dialog. Its interface is analogous to the Edit Object but the Save Scene as Object command allows you to create a new object for the *ISA* gallery and specify its own interface which is based on the interfaces of the constituent objects of the scene.

2. In the object definition, double-click the **material [Exp]** field under the **BUTTON_base** and **BUTTON.** Note that the object's name appears before the field name in the object interface list.

3. The procedure in item 2 creates two different fields in the button object interface that can be specified independently. However, it is sometimes convenient to associate different fields of the same type with the only interface element. The exercise in this step is just a sample of what you can do with the material fields. Let's change the object interface so as to associate the material fields of two sub-objects of button with the same interface element.

4. Remove the **BUTTON_base** from the object interface: right-click it in the Object interface list, and choose **Delete**.

5. Drag and drop the **material [Exp]** field under the **BUTTON_base** on the **BUTTON_material** in the Object interface list box. This procedure creates the multiple association of fields. Note that double-clicking **BUTTON_material** in the Object interface list highlights the material fields in gray for both objects alternatively (Find Next Associated Fields command from the right-mouse pop-up menu).

6. Rename BUTTON_base_material to material: right-click **BUTTON_base_material** and choose **Rename**.

7. In the "Object name" box type the name for your new object "**Button2**". Click **OK**. In the standard Save dialog select a folder to save the object. It is recommended that you create a new folder to save objects from lessons. The readymade object for this step can be found at "*…\ISA\Objects_Samples\Button2.wrl*".

Step 4. Replacing the object in a scene with its new version

1. Open the following scene, "*…\ISA\Samples\Sample2\Sample2.wrl*". Please note that *ISA* asks you whether or not to save the scene that is currently opened. Click No since the Button2 object from Step 3 has already been saved. We'll replace the Button1 object with its new version that exposes the *material* field.

2. Select the Button1 object.

3. Choose **Object > Replace Object**, then navigate to "…\ISA\Objects_Samples", select *Button2.wrl* and click **Open**.

4. In the Replace Object dialog, make sure the following options are chosen and the check box is turned on:

   Transform: **Keep current transform** (the default);
   Scale: **Keep current scale** (the default);
   Fixed point: **Bottom of bounding box** (the default);
   Keep child objects, abilities and animations (the default).

   Turn off **Use current object name**. This changes the name of the object to be replaced.

   You can get help on these options by clicking the Help button. For details on the Replace Object command and its options, see "Replace Object".

5. Click **OK** to replace the Button1object in the scene. Please note that this operation keeps the route from the *toggle* to *on* field in the Routing Diagram window and the value of *Enabled*. *ISA* always reconnect routes, if any, to the jacks of the new object whose names are the same as that for the old objects, and keeps the values for the same fields. The Button2 object now has the *material* field at the Properties tab that allows you to modify material types.

6. Try different materials and examine the scene in the Preview mode.

7. Save the scene. The ready scene for the current step can be found in *"…\ISA\Samples \Sample3\Sample3.wrl"*

Assembling a Scene

Creating, Opening, and Saving 3D Scenes

Create a New Scene

File menu > New.

To create a new scene:

- Click New on the Standard toolbar or choose New from the File menu. A new scene is created. All its parameters, such as grid spacing, etc. are set by default.

By default the newly created scene does not have any background image and landscape around it. To set the background scene, see “Use Background Schemes”.

Use Background Schemes

Tools menu > Options > Set Environment.

The newly created scene does not have any background image and landscape around it. *ISA* gives you several pre-built background schemes for your virtual world. These schemes include impressive landscapes like a summer, winter forest, or a mountainous area and much more.

To set the background scene:

1. On the **Tools** menu point to **Options** and click Set Environment.

2. Select a scheme or find another *.wrl* file in the Select Environment Object dialog box. Click Open.

Note:

- The background image or landscape appears in the scene whenever you create a new scene (New command from the File menu).

- You can use VRML files both from *ISA* and other VRML editors to create you own background schemes. The only requirement for the background scene is that the scene should contain the *ISA* objects, i.e. objects displayed in the Scene Tree window. All other geometry will not be shown in the newly opened scene. Nevertheless, you can always modify any *.wrl* file: open it in *ISA*, then save with the Save Scene as Object (Object menu) command, open a new *ISA* scene, add your new object to scene and save in the folder where you store the background scenes. You can also add any *.wrl* file by using the Add Object command. This command converts any geometry to an *ISA* object.

To clear the background scheme:

- Point to options from the Tools menu and click Clear Environment. The newly created scene appears in the scene without any background image and landscape.

Open an Existing Scene

File menu > Open.

To open a scene from a specified file:

1. Click the **Open** button on the **Standard** toolbar or choose **Open** from the **File** menu.
2. In the **Files of type** box, select the VRML Files item.
3. In the **Look in** box, click the location of the file.
4. Below the **Look in box**, double-click the folder you want.
5. Double-click the file name, or type it in the **File name** box.

A chosen scene is opened and is shown in the Perspective View window. The plan is displayed in the Plan window.

Note:

You can also open an existing scene by dragging its file onto *ISA's* title bar.

Import a Scene

File menu > Import.

Imports the VRML scene keeping the scene hierarchy. This command makes it possible to import a model consisting of several objects and to make all these objects accessible in *ISA*. You can add abilities and animations to any of the component objects of the model. You can also save any of the objects in the Object Gallery for future use or editing by using the Save Object command from the Object menu. When importing a VRML scene each Transform or Group node in it is converted to an *ISA* object. The object hierarchy remains unchanged. All animations and object's interactions are kept, if possible, but you cannot edit them in the imported scene.

Detailed information about importing:

This part of guide assumes that you are familiar with VRML97 and understand the structure of VRML files.

Every referenced node is converted to an *ISA* object with its own identifier if the source scene contains several instances of a Transform or Group node, for example:

```
#VRML V2.0 utf8
DEF  G Group{ }
USE G
```

or if two different Transform or Group nodes have the only child which is an instance of any node:

```
DEF   AnyNode Group {children […] }
      Transform { children [USE AnyNode] }
      Transform { children [USE AnyNode] }
```

In the cases when there are the Shape nodes and their geometry nodes are instances, the *material*, *texture* and *textureTransform* fields are exposed, if any, in the resulting *ISA* object:

```
DEF   T1 Transform
{
      children
      [
            Shape
            {
```

```
                    appearance Appearance {material NULL}
                    geometry DEF G  Box { }
                }
        ]
}
DEF   T2 Transform
{
        children
        [
                Shape
                {
                    appearance Appearance {material NULL}
                    geometry USE G
                }
        ]
}
```

All fields and events inside the Transform or Group node that have routes connected to them (except the SFNode and MFNode fields) are exposed in the newly created *ISA* object as properties or events with reconnected routes.

If the source scene contains two instances of a node and after importing these instances are in different *ISA* objects, instantiation is discarded and the nodes are duplicated. If one of the instances is a value of a Script node SFNode or MFNode field, the scene animation and any behavior may not work as expected.

If the source scene contains the Script node which in turn uses the SFNode or MFNode fields,

```
DEF S Shape
{
        geometry Cone { }
}

Script
{
        field     SFNode field_name USE S
        ...
}
```

ISA duplicates the instances and displays the Import message box:

"The following nodes referenced with USE statement in an SFNode or MFNode field of a Script node were duplicated in imported scene.

As result scene animation and behavior may not work as desired. + *[the list of duplicated nodes]*."

If *ISA* cannot determine how to connect routes to an SFNode or MFNode field, it displays the Import message box:

The following fields had routes connected to them. These routes could not be imported and were removed. As a result scene animation and behavior may not work as desired. + *[the list of removed routes: nodeNameId* **.** *eventInId]*

Save a Scene

File menu > Save, Save As.

To save a new unnamed scene:

1. Click the Save button on the Standard toolbar or choose Save from the File menu. A dialog box for the file name appears.
2. To save the document in a different folder, click a different drive in the Save in box, or double-click a different folder in the folder list.
3. To save the document in a new folder, click Create New Folder.
4. In the File name box, type a name for the VRML scene.
5. Make sure the VRML Files item in the Save as type box is selected to save the scene in the VRML 97 format.
6. Click Save.

To save an existing scene with the same name:

- Click Save on the Standard toolbar or
- Choose Save on the File menu.

To save a copy of a scene:

1. Choose Save as from the File menu. The Save Scene dialog box appears.
2. In the File name box, type a new name for the VRML scene.
3. To save the copy in a different folder, click a different drive in the Save in box, or click a different folder name in the folder list. To save the copy in a new folder, click Create New Folder.
4. Click Save.

Save Selection

File menu > Save Selection.

Saves the selected object with its sub-objects, if there are any. The Save Selection command allows you to save the part of the scene in a separate file to create reusable templates.

Prepare a Scene for Publication

File menu > Publish.

To prepare a scene for publication, *ISA* creates a new folder in which all data describing the scene is placed. This sub-folder will contain the scene's *.wrl* file and other resource files, for example, textures, sounds, pictures, etc. When published, all the file specifications in the current scene are converted to use relative addressing. You can also control the publishing procedure by specifying various options. After the folder is created you may upload the scene to your Web site or explore it on another computer.

To prepare the scene for publication:

1. Choose **Publish** from the **File** menu. The Publish Scene dialog box appears.
2. Click **Options** if you want to change the default settings for publication.
3. Enter the name of your scene *.wrl* file in the Name field.
4. Enter the full path name of a folder in the Location field. *ISA* creates a sub-folder with the name of the scene and places all the required data there.
5. Click OK.
6. If the scene you are publishing contains the links to the resource files that could not be copied to the folder you have selected for the scene, the Publish Warning dialog appears. You can copy the resource files manually using the URLs shown in this dialog. For details on these warnings, see “Publishing Warnings”.

Setting Publishing Options

Tools menu > Options > Publish.

Use this procedure to select the publishing options that are used when publishing your VRML scene. By selecting the corresponding option, you can control the internal structure of the VRML scene, select a folder where all scene resources should be stored, and significantly reduce the physical size of the final VRML (WRL) file.

Optimize scene. This selection provides the scene optimization and is recommended for most final VRML files. When on, the scene optimization is applied and other commands are also available. Please note that the optimization process removes *ISA* specific information which makes it impossible to edit a scene any further, so if you open the optimized scene, objects cannot be seen in *ISA* Scene Tree, Plan and other windows. Turn this option off to publish a scene for further editing on another computer.

> **Keep object names.** It is turned off by default. Turn this check box on if you want the Publishing Wizard to define each of the Transform node that corresponds the object with object's name from Scene Tree. This improves readability of the VRML file. When cleared, *ISA*, if it is needed, automatically generates numerical names to define the Transform nodes.
>
> **Add object’s type name to proto name.** It is turned off by default. When selected, it forms the identifiers of the PROTOs in the scene from the object-type name and globally unique identifier (GUID) of the PROTO. This improves the readability of the VRML file.
>
> **Optimize abilities, animations and system objects.** It is turned on by default. When selected, removes fields that are not referenced in routes or object interfaces. It is recommended to turn it off if you are going to manipulate objects in the VRML scene interactively from the external environment, for example, from scripts within HTML pages.
>
> **Unwrap single or static objects.** It is turned on by default. When selected the PROTO definition and the instantiation for single objects are replaced by the corresponding transformation node. For the several versions of the same geometric object, the transformation hierarchy of the DEF-USE syntax is applied.

Put all resources in subfolder. Lets you choose the folder where all resource files should be placed. When on, *ISA* stores resources in a separate folder with the name specified in the Subfolder name box. When off the resources are placed in the folder where the VRML scene is published.

Compress output file. Applies loss less compression to the file.

Remove extra formatting. Turn this check box on if you want the Publishing Wizard to reformat the VRML files by removing the indentations, unnecessary white spaces, and empty lines. Reformatted files use less disk space but usually lost in human readability.

Write default viewpoint. To store the current camera position as a viewpoint in the VRML file, select this check box.

Publishing Warnings

Resources That Could Not Be Published

Absolute paths to resources that could not be published to the resource folder, specified by the user, are listed in this window. Note that all URLs for resources in the published scene are relative. There are two reasons for the failure to publish a resource:

- There is no such a resource file at the specified path.
- The resource file couldn't be copied into the resource folder because, for example, the file already exists in this folder and the read-only attribute is turned on.

If you encounter this problem make sure that all the resources exist at the displayed paths. If there is no such resource file, put it there and publish the scene again. If the resource exists, make sure that there is no read-only file with the same name in the resource folder. If you find such a file there, remove it and publish the scene again or copy the resource manually.

Linked resources that should be published manually

The files referenced with the VRML Anchor nodes, i.e. the ability "Linked" and the Link object, are not published. Their URLs remain unchanged in the published file. They must specify either a global URL, for example, http://www.parallelgraphics.com/l2/template/images/toplogo.gif or a relative URL. If a link is relative, the scene developer must ensure the referenced file is in the corresponding folder.

Inlined resources that should be published to the same folder

VRML files that are referenced by the VRML Inline nodes in the scene and those that are referenced with the Inline and LOD objects are not copied to the folder for publishing because they can contain absolute links to the resource files. When publishing, absolute links to the inlined files are changed to relative ones, and they always point to the same folder where the scene is published. The scene developer must publish the files referenced by these links. To do that, open the inlined file in *ISA*, and then publish it to the folder where the scene have been published.

Undo/Redo Operations

Edit menu > Undo, Redo.

To reverse the last operation with an object, use the Undo button from the Standard toolbar, or select Undo from the Edit menu. Immediately after you undo an action, the Redo command becomes active, allowing you to restore what you have reversed.

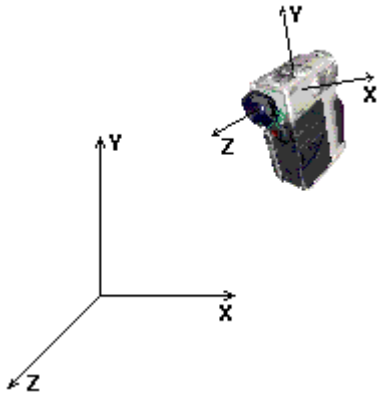
To reverse the action of the Undo command, use the Redo button from the Standard toolbar, or select Redo from the Edit menu.

There are 25 levels of Undo. Some actions cannot be undone (for example, moving a camera, switching navigation modes, and changing the Plan window parameters).

3D Navigation

Move a Camera in the Perspective View Window

Moving through a 3D space is similar to moving a camera. Think of a video camera that captures images in the real world and converts them into electronic signals for viewing on a screen. Your movements in the world continually position and orient that camera. The following figure shows a 3D world coordinate system represented as a set of X, Y, and Z axes (the Y axis is in the up direction), and the camera coordinate system is also shown with its own X, Y, and Z axes:



This concept assumes that there is a real person viewing and interacting with the VRML world. Moreover, the VRML author may place any number of viewpoints (or cameras) in the world – essentially interesting places from which the user might wish to view the world. Only one viewpoint may be active at a time. The following sections describe the mechanisms that *ISA* provides for navigating in a three-dimensional space.

- Using Viewpoints
- Moving Around: Walk, Fly and Examine
- Restore, Fit and Align
- Keyboard Commands
- Move a Camera in the Plan window

Using Viewpoints

A viewpoint describes a position and orientation for viewing the scene. The VRML author will probably want to guide the user to the best vantage points for viewing it. Not all scenes you open in *ISA* contain viewpoints but when they do you can use them. You can also add your own viewpoint to the scene by using the Viewpoint object from the System gallery.

To activate a viewpoint in the Design mode:

- Choose Go To Viewpoint from the pop-up menu, and then select a viewpoint from the list of the predefined viewpoints.

Note:

If there are no predefined viewpoints in a scene, the Go To Viewpoint command is grayed out.

Moving Around: Walk, Fly and Examine

Camera menu > Navigation mode > choose WALK, FLY, EXAMINE, or NONE.

Camera menu > Mouse mode > choose PLAN, PAN, TURN, or ROLL.

There are three main navigation modes: WALK, FLY and EXAMINE. You can select the navigation mode by clicking buttons on the Camera or vertical toolbar (it is not shown by default) in the Perspective View or choosing Navigation from the Camera menu. Each navigation mode may have several mouse modes: PLAN, PAN, TURN and ROLL. The combination of navigation mode and its mouse modes determines the possible camera motion and its orientation.

You can navigate with the mouse, the keyboard, or both mouse and keyboard. To move around a 3D world using the mouse:

1. Choose a navigation mode.

2. Position the pointer anywhere in the Perspective View window and press the left mouse button.

3. Move the mouse while holding down its left button. The *direction* in which you drag the mouse determines the camera motion.

4. Release the left mouse button to stop moving.

Note:

The distance that you drag the mouse determines the speed with which the camera moves. If you stop moving the mouse, the camera will continue moving until you release the mouse button.

To accelerate the camera's movement or rotation, press SHIFT, CTRL or SHIFT+CTRL when moving the mouse.

and or

and

Use WALK+PLAN to move in a horizontal plane.

Forward – move closer
Backward – move further away
Right – turn to the right
Left – turn to the left

Note:

Move the mouse forward or backward while holding down the Space key to turn upward or downward.
Move the mouse left or right while holding down the ALT key to move left or right.

and or

and

Use WALK+PAN to move left or right in a horizontal plane.

Forward – move closer
Backward – move further
Right – move right
Left – move left

Note: When you move the camera right or left the world will appear to move in the opposite direction.

and or

and

Use WALK+TURN to change the angle of the camera in a world.

Forward – turn upward
Backward – turn downward
Right – turn to the right
Left – turn to the left

and or

and

Use FLY+PLAN to move left or right.

Forward – move the camera forward towards its longitudinal axis
Backward – move the camera backward
Right – turn the camera to the right around its vertical axis
Left – turn the camera to the left around its vertical axis

Note: The camera's vertical axis may be inclined in a 3D space.
Move the mouse while holding down the ALT key to switch FLY+PAN.
Move the mouse while holding down the Space key to switch FLY+TURN.
Move the mouse while holding down the ALT+Space keys to switch FLY+ROLL.

and or

and

Use FLY+PAN to move up, down, left, or right within a single vertical plane.

Forward – move up
Backward – move down
Right – move right
Left – move left

Note: When you move the camera, the world will appear to move in the opposite direction.

and or

and

Use FLY+TURN to turn the camera.

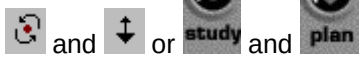
Forward – turn the camera upward around its horizontal axis
Backward – turn the camera downward around its horizontal axis
Right – turn the camera to the right around its vertical axis
Left – turn the camera to the left around its vertical axis

and or

and

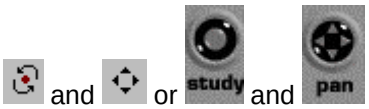
Use FLY+ROLL to incline the camera.

Forward – turn the camera upward around its horizontal axis
Backward – turn the camera downward around its horizontal axis
Right – incline to the left
Left – incline to the right

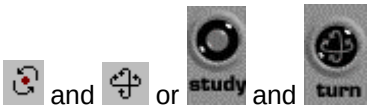


Use EXAMINE+PLAN to examine an object from various angles.

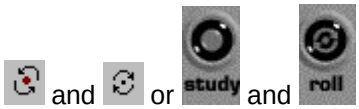
Forward – move closer
Backward – move further away
Right, Left – move the camera around the point defined by the pointer position where you press the left mouse button. If there is no face there, the scene coordinate origin is considered.



Use EXAMINE+PAN to move up, down, left, or right within a single plane of computer screen.



Use EXAMINE+TURN to examine an object from various angles. Select EXAMINE and TURN from the Camera toolbar or *Camera menu* > *Navigation* > *Examine*. Position the pointer over the object that you'd like to examine. Press the left mouse button. Drag the mouse to move the camera around the point on the object where you have pressed the button.



Use EXAMINE+ROLL to incline the camera around the point on the object where you have pressed the button.

Forward/Backward – the camera moves around the point on the object where you have pressed the button. If there is no face there, the scene coordinate origin is considered.
Right – incline to the left
Left – incline to the right

Use NONE to prevent navigation in the Perspective View window. This is useful if you manipulate objects directly in this window.

Use GOTO to move close to object in a world. Click GOTO on the toolbar in the Perspective View window, and then click on an object in the world. You'll move directly to it.

See also:

Adjusting a camera

Notes:

- You may not move a camera through walls while moving in the Perspective View window while the "collider" is on. To turn the collider on/off, select Use Collider from the Camera menu. You can also turn the collider on or off by clicking the button on the Camera toolbar.

- *ISA* automatically includes a light for the viewer in every world. The headlight always shines directly in front of the camera. To switch the headlight on, select Use Headlight from the Camera menu. You can also switch the headlight on and off by clicking the button on Plan toolbar or selecting Headlight from the right-button pop-up menu in the Perspective View window.

Restore, Fit and Align

There are three mechanisms that can help to re-orient a camera if you have lost your way in a world. Unlike the navigation tools, these buttons invoke predefined actions that take place as you click on them.

Use RESTORE to automatically return to the loaded world's original active viewpoint. This command is only available in Preview mode.

Use FIT to make the scene view fully visible in the Perspective View window.

or

Use ALIGN to position the camera's horizontal and longitudinal axes parallel to the scene horizontal (XZ) plane.

Adjusting a Camera

Camera menu > Go To Object From > Choose the appropriate command to view the selection.

While creating, editing, and assembling your scene, you can move the camera to a position, suitable for viewing object. The camera moves to fit the bounding box of the current object together with its children to the Perspective View window. Do one of the following to make an object or its group fully visible:

- Right-click the object in the Perspective View or object icon in the Scene Tree window, and then select Go To [object name].
- Double-click the object in Perspective View or its icon in the Scene Tree window.
- Select the object you want to view, and then click Go To Object on the Object toolbar or choose the appropriate command from the Camera menu.

To view object in your scene from Top, Bottom, Left, Right, Front, or Back:

- Select the object you want to view.
- Choose the appropriate button on the Object toolbar from the Go To Object group or choose one of the Go To Object From commands from the Camera menu.

Keyboard Commands

You can control the camera using keyboard commands. The functionality for arrow buttons corresponds to the movement of your mouse and depends on the navigation type and its option. Please note that the following description of keyboard commands is presented for the FLY+PLAN navigation.

- Arrow Up – move closer.
- Arrow Down – move further away.

- Arrow Right – turn to the right.
- Arrow Left – turn to the left.
- Arrow Up on the numeric keypad – move closer.
- Arrow Down on the numeric keypad – move further.
- Arrow Right on the numeric keypad – move to the right. If the EXAMINE mode – move the camera around the central point of bounding box of the selected object or 3D scene.
- Arrow Left on the numeric keypad – move to the left. If the EXAMINE mode – move the camera around the central point of bounding box of the selected object or 3D scene.
- 7 on the numeric keypad – turn the camera downward around its horizontal axis.
- 9 on the numeric keypad – turn the camera upward around its horizontal axis.
- 1 on the numeric keypad – incline to the right. Note: It is not available in the WALK mode.
- 3 on the numeric keypad – incline to the left. Note: It is not available in the WALK mode.
- Gray Plus – move up in the case of FLY mode.
- Gray Minus – move down in the case of FLY mode.
- 1 on the alphanumeric keyboard – incline to the right. Note: It is not available in the WALK mode.
- 2 on the alphanumeric keyboard – incline to the left. Note: It is not available in the WALK mode.
- 3 on the alphanumeric keyboard – turn downward. If in the EXAMINE mode – move the camera around the central point of bounding box of the selected object or 3D scene.
- 4 on the alphanumeric keyboard – turn upward. If in the EXAMINE mode – move the camera around the central point of bounding box of the selected object or 3D scene.
- Page Down – next viewpoint (only in the Preview mode).
- Page Up – previous viewpoint (only in the Preview mode).

Note:

- To accelerate the camera's movement or rotation: Press SHIFT, CTRL or SHIFT+CTRL and one of the above keys simultaneously.
- The navigation keyboard commands are available only when the Perspective View window is in focus. Just click somewhere in the Perspective View before using the keyboard.

Move a Camera in the Plan Window

To move a camera along a plan:

1. Position the pointer over the Pinocchio icon. The pointer looks like a wide cross.
2. Drag the Pinocchio icon in the desired direction.

Note:

- You may navigate the camera through walls while moving in the Plan window.

To change a camera's orientation:

1. Position the pointer over the Pinocchio icon tip. The pointer looks like a cross.

2. Rotate the Pinocchio icon tip with your mouse (yawing in the Top view, pitching in the Front or Left view).

To place a camera in a required position:

- Double click the desired position in the Plan window. The camera moves to that location.

Setting Perspective Window Options

Right-click in the Perspective Window > choose Preferences.

The following options are available from the right-button pop-up menu of the 3D View window. Please note that the list of commands in the pop-up menu is different and depends on whether *ISA* is in the previewing or editing mode. In the editing mode the pop-up menu duplicates commands that are commonly used and modifies the Viewpoint command to Go To Viewpoint.

- **Viewpoints/Go To Viewpoint**. Activates a list of the viewpoints in the scene. In the editing mode this shows both viewpoints that have come with the opened VRML scene and the viewpoints added to scene by using the Viewpoint object from the System gallery. At first the descriptive name of the viewpoint is shown and the object's name in brackets.

- **Headlight**. The headlight always shines directly in front of the camera. You can switch the headlight on and off.

- **Navigation**. You can select a navigation mode.

- **Speed**. Controls the rate at which a camera moves trough a world.

- **Show Toolbars**. You can show or hide toolbars.

- **Preferences**. Modifies settings for Cortona VRML Client (used as a 3D viewer) such as screen appearance, a renderer mode, and other options. You can also configure Cortona options according to your preferences by choosing Tools>Options>Perspective View. The Perspective View Options dialog box is then displayed.

General tab at Preferences

- **Loading**. The Show progress check box determines whether the current state of the loading process is shown. If the Wait for all resources is selected, the scene will not be shown until all resources are loaded. If it is cleared, the scene's geometry will be shown immediately after the main *.wrl* file loading.

- **Appearance**. You can change the background color of the Cortona 3D window: Click the Background color box at the General tab, click the desired color from the palette, and then click OK. VRML authors can also control a color that simulates ground and sky.

- **Display frame rate**. Allows you to display the frame rate on the status bar.

- **Console mode**. Shows or hides the VRML console containing errors or warnings. The ConsoleMode attribute is set to Autolaunch by default. This shows the VRML console if errors or warnings occur.

- **CPU load**. Determines the degree of acceleration for Cortona renderer. This allows you to specify the processor usage in the range from 0 (minimum frame rate, maximum the processor idle time) to 100 (maximum frame rate).

Renderer tab at Preferences

Allows you to select a rendering mode. To draw the 3D image, Cortona provides two hardware renderers: OpenGL and DirectX. If your system has hardware acceleration for either OpenGL or Direct3D (note: you must have DirectX 5 or greater installed), choose the appropriate renderer. Using hardware renderers can introduce limitations. Select a renderer to compare the performance and visual quality of hardware versus software rendering and set your preferences accordingly.

Renderer options

- **Dither colors if needed**. Controls whether Cortona dithers while rendering. Dithering improves the quality of rendering, but may lower performance.

- **Motion blur effect**. Apply a motion blur effect. This actually makes an image seem a little blurred in motion.

- **Wireframe rendering mode**. The object appears to be outlined with wires rather than solid.

- **Anti-aliasing**. The technique is for smoothing out jaggies in showing curves on a computer monitor. There are two different modes: Idle-time and Real-time. If the Idle-time option is on, it is applied only for a static scene. Whereas selecting Real-time initiates smoothing in the viewer movement but also results in a decreasing frame rate.

- **Do not render textures**. To turn the textures on or off.

- **Limit textures size**. All textures are optimized for speed.

- **Optimize texture for quality**. All textures are optimized for quality. Note that if you select both Optimize texture for speed and quality, the renderer uses the available resources to optimize speed and quality simultaneously.

- **Use textures mip-mapping**. When the scene contains acutely angled polygons that disappear into the distance, this mixes low and high resolution versions of the same texture to reduce the jagged effect.

- **Advanced alpha blending**. Alpha blending is the name for the alpha channel control and it's used to simulate effects such as placing a piece of glass in front of an object so that the object is completely visible behind the glass or unviewable or something in between. Advanced alpha blending turns a particular rendering method on the better quality of translucent faces. The processor should support MMX additional instructions.

- **Automatic level of detail**. Automatically reduces the number of polygons which are displayed for any object at large distances (tiny objects on computer screen) from the viewer.

- **Extended Z-buffer**. If the check box is selected, this sets the 32-bit Z-buffer for the software renderer (R98). The default is 16-bit.

- **Phong lighting model**. A particular method for computing the apparent color of a face at a particular point. It improves the quality of lighting but at a lower performance. Note that this option is available only for R98 Renderer in the Intel Pentium III optimization mode.

- **Realistic fog**. A method for computing the color for every pixel in the frame buffer to simulate fog. It improves the quality, but lower performance.

- **Optimization**. Turns the optimization rendering mode on or off (if available).

- **Renderer version**. This is accessible only for Direct3D renderers. Allows you to select a Revanche DX5 rendering mode in the case you have DirectX 7 or a later version installed.

Navigation tab at Preferences

- **Navigation mode**. This allows you to select the navigation mode from a list of available navigation paradigms.
- **Travel speed**. Sets the rate at which the viewer travels through a scene. The following are the typical values for the rates, in metres per second: Slowest=0.0625, Slower=0.25, Normal=1, Faster=4, Fastest=16. If the speed field of the NavigationInfo is specified in the VRML file, the corresponding speed factors are multiplied.
- **Animate viewpoint**. Sets the Viewpoint transition rules that specify how Cortona interprets the transition from the old viewpoint to the new one. Auto – defined by the jump field of the Viewpoint node of a new location. Always – a jump with the transition effect. Never – instantaneous transition.
- **Collision detection**. By default, Cortona will allow you to pass through objects in your path. To prevent from passing through objects, select Always, in the Collision detection box at the Navigation tab. Select Auto to use the collision method specified by the VRML author.
- **Headlight on**. Mark the check box to turn the headlight on.
- **Show hidden viewpoint**. This allows you to see all viewpoints of the scene in the VIEW list including those that do not have a description (a Viewpoint's description field is empty).

Using Objects

Overview of Using Objects

In *ISA*, "objects" mean objects added to the VRML scene from the galleries and these are the building blocks of a scene. Objects can describe both primitive geometry like boxes and spheres and more complex geometry such as characters or objects with animations, menus, and so on. Geometric objects are renderable. A scene can also contain nonrenderable objects such as lights, cameras, VRML node-like objects, logical helpers, called system and logical objects. Objects can be grouped together to form a family tree analogy. This describes the relationship between objects linked together in a hierarchy. A group can have any number of members, or children, which can be either objects or other groups. You can take a readymade object from the Object gallery. This object can be placed into the scene. You can change the size, position, and the orientation of geometric objects, set an animation sequence for it and add or change its properties.

This section presents the following topics:

The Scene Tree
Place Objects into a Scene
Select Objects
Delete Objects
Rename Objects
Drop Objects
Align an Object
Move Objects to Another Hierarchic Level
Cut, Copy, and Paste Objects
Replace an Object
Save an Object
Save Selection
Save Scene as Object
Edit Object
Insert Objects from Scene
Find an Object

Freeze an Object
Wrap by Group

See also:
Import Scene
Creating an ISA Object

The Scene Tree

The Scene Tree presents information about a scene structure and provides access to tools to adjust the hierarchical linkage between objects. By linking one object to another you create a parent-child relationship. Transformations applied to the parent are also transmitted to the child. By linking objects to both parent and child objects you can create complex hierarchies to create, for example, complex motions or simulate jointed structures. The Scene Tree is very similar to a genealogical tree. Each tree node represents an object. Each node may hold any number of child nodes (sub-objects). Each node, with the exception of the primary nodes, has a parent. Primary nodes are joined together.

Object images in the Scene Tree:

- The represents ordinary (static) VRML objects or objects with internal behavior properties. Some of these objects can be connected with each other in the Routing Diagram window.

- The icon indicates a frozen object that you cannot select.

The relations between *ISA* objects are defined in the following way:

- When an object is resized or moved (rotated), all its sub-objects are resized by the same coefficient or moved (rotated) by the same distance (angle).

- If an object is deleted, all its sub-objects are also deleted.

- Mutual subordination of the objects in a scene can be arbitrarily changed.

Note:

- The scene is your world, and like the real world we can't move, resize, or turn it. This means that there are no manipulators for the whole scene.

- Objects appear in the Scene Tree under their gallery names which you can change later on.

The most useful commands available in the Scene Tree are described in the following topics:

Adjusting a Camera
Select Objects
Move Objects to Another Hierarchy Level
Cut, Copy, and Paste Objects
Wrap by Group
Rename Objects
Freeze an Object

Place Objects into a Scene

To place an object from the Object gallery in a scene:

1. Click a tab in the Object gallery then scroll the gallery to find the object you need. Repeat this step with other tabs till the object is found.

2. Drag the object's thumbnail to the scene with the mouse.

You can drag and drop objects into the Plan, Perspective View, Scene Tree, and Routing Diagram windows:

- **Plan window.** An object is located in the place where you have dropped it. The center of its bounding box is set on the Plan at a point where the mouse button was released while dragging. The third coordinate of the bounding box center, which cannot be defined from the current plan view, is by default set in such a way that the object appears with its bottom face at zero level of the vertical axis if you have dropped the object on the Top view, and with its bounding box center at zero for other projections.

  Using the Top Plan view you can choose two different ways to place objects in the scene:

  On Zero Level. The object appears with its bottom face at zero level of the vertical axis.

  On Top Face. The object attaches with its bottom face to the top of underlying object. The dropped object becomes a child of the object that is associated with the face upon which you drop the object. If there is no underlying object, the dropped object appears with its bottom face at zero level of the vertical axis.

  You can choose the way to place objects in the scene by specifying the Drop method under Modes in the Plan Properties dialog (*Tools menu* > *Options* > *Plan*).

- **Perspective View window.** An object may be located anywhere in a scene. The dropped object becomes a child of the object that is associated with the face upon which you drop the object. The object attaches to the face with its own bottom face. This guarantees, for example, that when a chair is placed on the floor, it is directly on the floor, not suspended above it or passing through it. If the object is placed into a vacant area of the scene, the coordinates of the center of its bounding box are set 10m from the camera.

- **Scene Tree window.** The object becomes a child of the object upon which it was dropped. The center of its bounding box is located in the origin of the VRML world coordinate system.

- **Routing Diagram window.** The object is located in the scene coordinate origin. If the object has no incoming or outgoing events, it is invisible in the Routing Diagram window.

Objects placed according to the above methods are eligible for all object operations, such as rotation, resizing, renaming, etc.

Note:

Object files in the gallery store information about the object's orientation including the direction of the vertical axis of its coordinates. An object in the gallery is displayed in such a way that its orientation coincides with the orientation it will have when placed into a scene. When dragging an object to a face in the Perspective View window, the object "falls" in such a way that its vertical axis coincides with the target face's normal vector. Therefore, an object such as a chair oriented in the gallery with its legs downward is always placed on a face with its legs directly touching the face.

In order to move an existing VRML object from a file to a scene, you must do one of the following:

- Choose **Add Object** from the **Object** menu and find an object in the Add Object dialog box. Click OK. A new object is located with its pivot point at the origin of the VRML world coordinates. If any object in the scene is selected, the object you add becomes its sub-object, and appears at the origin of the coordinates of the parent (selected) object.

- Drag the file you want to the desired *ISA* window. You can drag and drop objects into the Plan, Perspective View, Scene Tree and Routing Diagram windows. The object appears in the scene the same way as if you were dragging it from the Object gallery.

To copy an object using the Scene Tree window:

- Hold down the CTRL key then drag the object you want to the Perspective View, Plan or Scene Tree window. The copy of the object appears in a scene.

To move an object using the Scene Tree window:

- Drag the object from the Scene Tree to the Perspective View window or Plan. The object then moves to its new location.

Select Objects

To manipulate an object you should select it in the Perspective View, Scene Tree, Plan or Routing Diagram window.

To select an object in the Perspective View window:

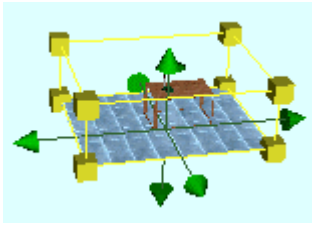
- Click the object. The selected object or the hierarchic group of objects is marked with a yellow bounding box.

Note:

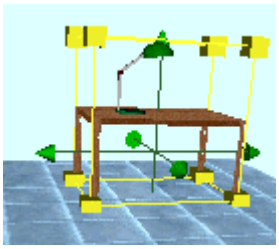
If the object is a member of the hierarchic group of objects in the Scene Tree window as shown on the following figure,

there are two ways to select objects. To move up or down through the levels of hierarchy, click the Select Lowest or Select Highest button on the Object toolbar. You can also find these buttons at the Tools menu > Options.

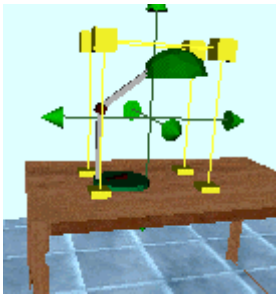
Select Highest Clicking the Lamp object in the Perspective View window results in the selection of all these objects:



By clicking the Lamp object again, you select the objects one level down in the hierarchy (a table and a lamp):



When you click the Lamp for the third time, this selects the lowest node in the hierarchy (a lamp):



Select Lowest When you click the object in the Perspective View window, this selects the object with its sub-objects. You can move up a level by clicking on the object again.

To select an object in the Scene Tree window:

- Click the object's name or icon. The name of the selected object is highlighted. The bounding box that surrounds the selection and all its sub-objects appears in the Perspective View window.

To select an object in the Plan window:

1. Make sure one of the buttons from the object manipulation group on the Plan toolbar is pressed.

2. Move the pointer into the Plan window and point to any line of the object you want to select in the Plan window. The pointer should look like a small plus.

3. Click the line of the wireframe projection an object. It turns red.

Note:

If the object is a member of the hierarchic group of objects in the Scene Tree window, there are two ways to select objects (see the previous note for details). To move up or down through the levels of hierarchy, click the Select Lowest or Select Highest button on the Object toolbar, or choose the appropriate command from Options on the Tools menu.

To select an object in the Routing Diagram window:

- Click the object's name at the top of the icon.

Delete Objects

Edit menu > Delete.

To delete an object:

1. Select the object to be deleted.

2. Choose Delete from the Edit menu or click on the Standard toolbar or press the Delete key.

Rename Objects

To rename an object do one of the following:

- Click the highlighted object name in the Scene Tree window. An edit box for the object name appears. Enter the new name and press ENTER.
- Select the object you want to rename, click the Properties tab in the Properties/Events window, click on the right of (Name) and type a new name.

Drop Objects

Object menu > Drop.

To place an object in the scene on the top of the other underlying face:

1. Select the object you want to drop.
2. Click the Drop button on the Object toolbar. The object attaches with its bottom down to the first underlying horizontal face and becomes its sub-object.

Note:

When you drop an object to a face, the object "falls" in such a way that its vertical axis coincides with the target face's normal vector. An object such as a chair oriented in the gallery with its legs downward is always dropped on a ceiling with its legs upward.

Align an Object

Object menu > Align.

When you rotate an object in the scene, you may need to align the object's axes parallel to the axes of the VRML world, or the parent object (the object one level up in the hierarchy).

To align an object:

1. Select the object you want to align.
2. Choose Align from the object menu or click the Align button on the Object toolbar.

To align the object directly in the Perspective View window:

- Right-click the object you want to align in the Perspective View window and then click Align in the appeared pop-up menu.

Move Objects to Another Hierarchic Level

To move an object to another hierarchic level in a scene do one of the following:

- In the Scene Tree window drag its name to the name of the object you want to make a parent – drag until the parent's name becomes highlighted. If you want the object to be at the top of hierarchic level, drag its name to the vacant place below all icons in the Scene Tree window.
- Select the object you want to move, in the Properties/Events window click the Properties tab, click (Owner), click it the button, and select the new parent's name or (None) if you want the object to be at the top level.

The Scene Tree displays objects in the same order as they will appear in the VRML file. By dragging the object's name with the SHIFT button pressed, you can change this order and as a result accordingly change the object's hierarchy level. It is useful in cases when you want the one of the viewpoints, fogs or backgrounds (these correspond to bindable nodes in VRML) to be activated first when you start the scene in the VRML browser. According to the rules in VRML, the first encountered bindable node is bound by the VRML browser when it reads a file.

To move an object to another place in the finished VRML file:

- In the Scene Tree window drag its name with the SHIFT button pressed to another place. The name of the object that will be the parent's name is highlighted. The horizontal line indicates a new position for the selection.

Cut, Copy, and Paste Objects

Edit menu > Cut, Paste

You can cut or copy an object onto the Clipboard and paste it into another place in your scene.

To cut or copy the object onto the Clipboard:

1. Select the object you want.

2. Click the Cut button on the Standard toolbar or choose Cut on the Edit menu to cut the object to the Clipboard – or

   Click the Copy button on the Standard toolbar or choose Copy from the Edit menu to copy the object to the Clipboard. The object and all its sub-objects are copied onto the Clipboard.

To cut or copy the object directly in the Perspective View, Plan and Scene Tree windows:

- Right-click the object you want to cut or copy in the Perspective View, Plan or Scene Tree window, and then click Cut or Copy in the pop-up menu. The object and all its sub-objects are copied onto the Clipboard.

To paste the object from the Clipboard, you must do one of the following:

- Click Paste on the Standard toolbar or choose Paste on the Edit menu to paste data from the Clipboard to the scene. If any object in the scene is selected, the object you paste becomes its sub-object, and appears at the same place in the scene as the original object.

- Right-click the place in the Perspective View or Plan window where you want to paste the object. The pop-up menu appears. Click Paste. The object will be inserted into a scene so that the object's pivot point will appear at the click point on the plan. In the Perspective View window, the object will be inserted into a scene so that the bounding box center appear at the click point, and if the object is placed into a vacant area of the scene, the coordinates of the center are set 10m from the camera. If you right-click any object in the Perspective View, the object you paste becomes a child of the object on which it was dropped, and it attaches to the face with its own bottom face.

- Right-click the place in the Scene Tree window where you want to paste the object. The pop-up menu appears. Click Paste. The object appears at the same place in the scene as the original object.

To replace an object currently selected onto an object stored on the Clipboard:

- Choose the Paste and Replace command from the Edit menu, in the Replace Object dialog, specify the method by which the new object will be inserted and then click OK.

To copy an object using the Scene Tree window:

- Drag the object with the CTRL key pressed to the Perspective View or Plan window. The copy of the object appears in a scene.

Replace an Object

Object menu > Replace Object.

This command allows you to replace the selected object in your scene from the external file.

To replace an object:

1. Select the object you want to replace.
2. On the **Object** menu, choose **Replace Object**.
3. In the **Look in** box, click the location of the VRML file.
4. Below the **Look in** box, double-click the folder you want.
5. Double-click the file name, or type it in the File name box, or click Open.
6. In the Replace Object dialog, specify the method by which the new object will be inserted, click OK.

See also:

Replacing an object in the scene with the object from the Object gallery

To replace an object currently selected onto an object stored on the Clipboard

Save Object

Object menu > Save Object.

The Save Object command helps you to save the selected object from the opened scene.

To save an object:

1. Select the object you want.
2. On the **Object menu**, choose **Save Object**. A dialog box for the file name appears.
3. In the **File** name box, specify an object name under which the object will be displayed in the Object gallery.
4. Choose a gallery name where you want to save the object. Note that if you select the Objects folder, your new object appears at General tab.
5. Click **Save**.

Note:

- Only predefined properties and events will be saved with your new object.
- Animations and abilities associated with the object in the scene cannot be saved.
- This command saves only the selected object and does not save its sub-objects, if there are any.

Save Scene as Object

Object menu > Save Scene as Object.

Use this command to convert an *ISA* scene to the file format of an *ISA* object; save it under a different filename; and choose the fields and events that will be visible in the Routing Diagram or Properties/Events windows whenever this new object is placed in a scene. When everything is ready, bring all the resource files together and optimize your object by setting the Publish option in the Save Scene As ISA Object dialog.

The steps that you take to save your scene as a new *ISA* object depend on what fields and events you want to expose.

To save the current scene as a new *ISA* object:

1. Click the Save Scene as Object button on the Object toolbar or choose Save Scene as Object from the Object menu. The Save Scene As ISA Object dialog box appears.

2. In the Object Name box, type a name under which the object will be displayed in the Scene Tree of *ISA*.

3. In the "Object definition" list, double-click the fields or events you want to be accessible in your new object. They appear in the "Object interface" list in the lower left.

4. Change the name of the event or field if needed. To do that, click it twice in the "Object interface" list, type the new name, and then press ENTER.

5. For another procedures, see “Object Interface Control”.

6. Click OK then choose a gallery name where you want to save the object and object name, under which the object will be displayed in the Object gallery. Note: if you select the Objects folder, your new object appears at the General tab.

Note:

- Items 3 – 5 are optional since the object interface can be empty.

- Gallery’s name is the name of a sub-folder of the default *ISA* object folder. For example, if *ISA* is installed in C:\ISA, its default object folder is C:\ISA\Objects. All objects files from the Basic gallery are placed in C:\ISA\Objects\Basic.

- The *ISA* installation package contains examples for scenes and objects that provide a simple way to learn how to use Save Scene as Object command. Look in the folders *ISA**Samples* and *ISA**Objects*_*Samples* for scenes and objects respectively. These lessons contain step-by-step exercises which teach you how to create new *ISA* objects from the objects of other VRML editors and *ISA* itself:

  Lesson 2. Creating a New Animated Object
  Lesson 3. Editing an Object

Edit Object

Object menu > Edit Object.

This command allows you to choose for the selected object in a scene the fields and events that will be visible in the Routing Diagram or Properties/Events windows. Its functionality is analogous to the Save Scene as Object command but all object modifications are available when you assemble a scene.

Note: Despite the fact that *ISA* gives you the ability to create and save custom object interfaces and configurations, we don’t recommend that you edit the finished objects that are shipped with the program in the Characters, Control, Image, Logical, Navigator, System, Text and Transform galleries.

To edit the object interface:

1. Select the object.
2. On the **Object menu**, choose **Edit Object**. The Edit Object dialog box appears.
3. In the "Object definition" list, double-click the fields or events you want to be accessible in your object. They appear in the "Object interface" list in the lower left.
4. Change the name of the event or field, click it twice in the "Object interface" list, type the new name, and then press ENTER.
5. For another procedures, see “Object Interface Control”.
6. Click OK.

Insert Objects from Other Scene

Edit menu > Insert Objects.

The Insert Objects from Scene command (Edit menu) inserts all objects from the selected scene to your current scene. When you select this command, the Insert Scene dialog appears, prompting you to open or find a scene from which to extract all objects.

Find an Object

Edit menu > Find Object.

To find an object using its name:

1. On the **Edit** menu, click **Find Object**.
2. In the **Object name** box, enter the object’s name that you want to search for.
3. Select any other options that you want.
4. Click **Find Next**.
5. To cancel a search in progress, press ESC.

Search options:

Match case. Select this check box to distinguish between uppercase and lowercase characters.

Match whole names only. Select to search for the whole names of object in the Scene Tree window.

Use wildcards (* and ?). You can use wildcards to specify search criteria. Type a question mark (?) to match any single character, or type an asterisk (*****) to match any number of characters. For example, **L?d** finds "Lod" and "Led"; **L*d** finds "Lod" and "Lead."

Freeze an Object

Right-click the object in the Scene Tree window > choose Freeze [object name].

It is often useful to have an object that cannot be selected to prevent it from moving, rotating, or being removed and changing its hierarchy level when editing a scene. Note that the Freeze command allows you to select children of the frozen object.

To Freeze/Unfreeze an object:

- Right-click the object icon in the scene tree window and then select Freeze [object name] or Unfreeze [object name] from the pop-up menu.

Wrap by Group

Right-click the object in the Scene Tree window > choose Wrap by Group [object name].

The Wrap by Group command allows you to create additional level of hierarchy in a scene. The Group object becomes the parent of the selected object and uses the same transform. The child object does not have any transformation relative the Group object. It is useful in cases when you use objects from the Transform gallery.

Using Transforms

A transform is an adjustment of an object's position, orientation, or scale, relative to the 3D world coordinate system in which you are working. You can apply three types of transform to an object:

- Position
- Rotation
- Scale

To use a transform, select one of the seven plan projection buttons, then select one of the three coordinate systems that meets your needs, and click one of the four transform buttons on the Plan toolbar, or choose a transform from the pop-up menu in plan. You then apply the transform to a selected object using the mouse, a type-in dialog, or both. Moving translates an object in the 3D world. Rotation changes the orientation of an object. Scaling changes the size of the object.

These sections contain explanations of various manipulators and commands on how to transform objects, and how to animate your transforms:

Manipulate objects using Plan
Manipulate objects in the Perspective View window
Placement

Placement

Object position, orientation and size are set with the object manipulators both in Plan and Perspective View. However, it is sometimes convenient to manipulate the object by setting the numerical values:

1. Select the object.
2. Click the Placement tab in the Properties/Events window.
3. Click the Absolute or Relative field to enter the appropriate mode.
4. Enter the desired value and click OK.

Absolute

allows transformation in the local coordinate system of the parent object. If the object is at the top hierarchical level, the VRML coordinate system is assumed.

Relative

allows transformation from the current position of the object. A transformation is applied about the local coordinate axes of the selected object. Here these axes would be rotated together with the object.

Translation

to move an object by specified amounts in the X, Y, and Z directions, in metres.

Scale

specifies a scale which can be a non-uniform. If scale values in the X, Y, and Z directions are the same, changing the scale in X results in uniform scale, whereas Y and Z fields specify a non-uniform scale for the corresponding axis.

Rotation

to rotate around X, Y, and Z axes, in degrees.

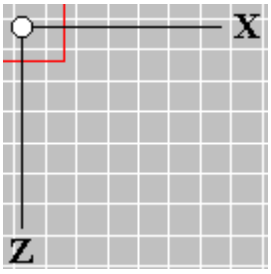
Center

allows you to move the Object Manipulators moving point (center).

Manipulate objects in the Plan window

Using the Transform Manipulators

Select an object > Plan toolbar > Click any transform button



The transform manipulators are shown as coordinate axes. These let you quickly choose one or two axes when transforming a selection with the mouse. You choose an axis by placing the mouse pointer over any axis, then dragging the mouse to transform the selection along that axis.

To select an object in plan:

- Position the pointer in the Plan window on any line of an object you want to manipulate. The mouse pointer changes to a small plus icon:

- Click a plan's line associated with the desired object. The object becomes selected (the default color is red) and the coordinate axes appear nearby.

Note:

You can also select an object both in the Perspective View and Scene Tree window.

Moving and Rotating Objects

Plan menu > Manipulation Mode > choose Move Object or Rotate object.

To move an object

- Make sure the object is selected, click Move on the Plan toolbar, choose an axis by placing the pointer over it, the pointer changes to , press the left mouse button and then drag the pointer along the selected axis to apply the transform.
- Or you can position the pointer on any line of an object or over its pivot point. The pointer changes to . You can now move the object. The direction of the movement is determined both by your mouse and by the current transformation coordinate system. In addition, the Axis Constraint buttons limit movement along one or two axes.

To rotate an object

- Make sure the object is selected, click Rotate on the Plan toolbar, choose an axis by placing the pointer over it, the pointer changes to and the corresponding Axis Constraint button becomes pressed, press the left mouse button, and then drag the pointer across the selected axis to apply the transform.
- Or you can position the pointer on any line of an object or over its pivot point and the pointer changes to . You can now rotate the object. The direction of the rotation is determined both by your mouse and by the current transformation coordinate system. In addition, the Axis Constraint buttons limit the rotation about the selected axis.

Scaling Objects

Plan menu > Manipulation Mode > choose Uniform Scale objects or Non-uniform scale object.

Scaling changes the size of an object. The Scale Object transform button is a flyout menu that provides two types of scale. The type of scale visible on the Plan toolbar corresponds the type that is selected in the pop-up menu, which is accessed by right-clicking a selected object.

Use the Uniform scale to select equally along all three axes. The center of the scale is determined by the object's pivot point. Specifying the reference coordinate system (Coordinate System group) has no effect on uniform scaling, whereas the axis constraint buttons immediately activate the non-uniform scale.

Use the Non-uniform scale to scale the selection differently along the three axes or to any two axes. The Axis Constraint buttons determine the axis or axes along which the scaling occurs. The Transform Coordinate system determines the direction of the scaling and the object's transformation center determines the center from and to which the scaling takes place.

To uniformly scale an object

- Select an object, choose Plan>Manipulation Mode>Uniform Scale Object or click the button on the Plan toolbar, place the pointer on any line of the object in Plan, the pointer changes to . You can now scale the object in a single mouse action.

To use non-uniform scale

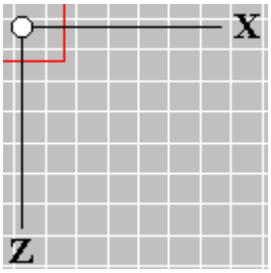
- Select an object, make sure the Scale Object button is pressed on the Plan toolbar, click one of the Axis Constraint buttons, place the pointer on any line of the object in Plan, the pointer changes to . You can now scale the object in a single mouse action.

- Or you can choose an axis by placing the pointer over it, the pointer changes to and the corresponding Axis Constraint button appears pressed, press the left mouse button and then drag the pointer along the selected axis to apply the non-uniform scale.

Note: Avoid applying the non-uniform scale at the object level. Non-uniform scaling is applied as a transform and changes the axes of the object, so it affects other object properties. It also alters the properties passed hierarchically from parent to child. When you perform other operations on the object, such as rotation and other positioning operations, you may not get the results you expect. To recover from these problems, use the Scene Tree window to change the scene hierarchy.

Choosing a Transform Center

Plan menu > Manipulation Mode > choose Move Object Pivot.



The transform center (also called the object's pivot point) affects scale and rotation transforms, but has no effect on position transforms. By default, the object's pivot point locates at the origin of the local coordinate system of the object.

To change the location of object's pivot point:

- Make sure the object is selected, click Move Object Pivot on the Plan toolbar, choose an axis by placing the pointer over it, the pointer changes to , press the left mouse button and then drag the pointer along the selected axis to apply the transform.

- Or you can position the pointer on object's pivot point, the pointer changes to . You can now move the pivot point. The direction of the movement is determined both by your mouse and by the current transformation coordinate system. In addition, the Axis Constraint buttons limit movement along one or two axes.

Correcting a Transform Center

Object menu > choose Correct Pivot.

The Correct Pivot command places the transform center of the selected object in the center of its bounding box.

Using the Transform Type-In

Select object manipulation mode on the Plan Toolbar > Right-click the selected object in the Plan window > choose Transform Type-in.

Transform Type-In allows you to enter precise values for move, rotate, and scale transformations, and also change a position for a Transform Center. You can use the Transform Type-In with objects in plan that can display an axis tripod. To begin, select the object, choose the transformation mode by clicking one of the four transform buttons on the Plan toolbar or choosing a transform from the pop-up menu in plan, and then select *Transform Type-in* from the right mouse menu in plan. The title of the appeared dialog reflects the currently selected transformation. If Move Object is active, the dialog's title is Move Object and its boxes affect translation. If the Rotate Object is active, its title is Rotate Object, and so on. All values entered are reverted to zero after each operation. If you type 45 in the Angle in degrees text box and then press ENTER, the object is rotated 45 degrees from its previous position, and the dialog box is closed. In addition, the coordinate axes for the transformation are determined by currently selected reference coordinate system (*Plan menu* > *Coordinate System*). The Axis Constraint buttons have no effect on the transformation.

INTERFACE

- Translation dialogs (move an object or its pivot point). X, Y, and Z: These fields display and accept entry for offsets of the values of position along all three axes.
- Rotation dialog. The Around Axis group allows you to select the axis to be used to perform rotation. The Angle in degrees text box accepts entry for the rotation, in degrees.
- Scale dialog. X, Y, and Z: These text boxes display and accept entry for the scale operation differently along all three axes. The current position of the Transform Center determines the center from and to which the scaling takes place.

The Axis Constraint Buttons

Plan menu > Restrict to > Click any Axis Constraint button to limit transformation.

These buttons on the Plan toolbar let you specify one or two axes about or along which the transform takes place. Only one of six Axis Constraint buttons is active at a time. When a button is turned on, transforms are constrained to the axis it specifies. For example, if you turn on the "X Axis" button, you can rotate an object only about the X-axis of the current coordinate system. Please note that the restriction you choose affects only transformations when you move, rotate or scale an object by placing the pointer on an object’s line and then moving the mouse. The first three buttons constrain the transform to a single axis. The fourth button is a flyout menu and has additional commands in the sub-menu. These buttons specifies double-axes combination. To display the flyout menu and activate the item, press and hold the flyout, position the pointer over the item you need and then release the mouse button.

Manipulate objects in the Perspective View window

You can change the object’s position, scale and orientation directly in the Perspective View window with the help of object manipulator controls. This is useful when:

- You need to position and set the size of the object in a 3D space relative to the other scene items and view the result directly in the Perspective View window.
- You need to place the object onto the specific face of the other scene item, for example, on the ground landscape surface or on the floor of the virtual house.

To display the object manipulators in the Perspective View window, select the appropriate object in the Perspective View window, Plan window or Scene Tree window.

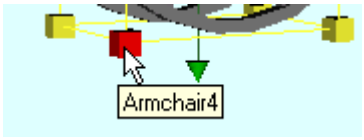
Using the object manipulator controls in the Perspective View window you can:

- Rotate an object
- Scale an object
- Move an object to another place and snap it onto another face in the scene

Object Manipulator Controls in the Perspective View Window

The object manipulator controls in the Perspective View window are shown on the following pictures. There are two types of controls:

Object rotation controls: green cones at the end of the axes crossing the object center. When you position the mouse pointer over the rotation control, the cone is colored in red and object name appears on the screen.



Object resizing (scale) controls: yellow boxes in each bounding box vertex. When you position the mouse pointer over the scale control, the box is colored in red and the object name appears on the screen.

To move the object rotation controls relative to an object:

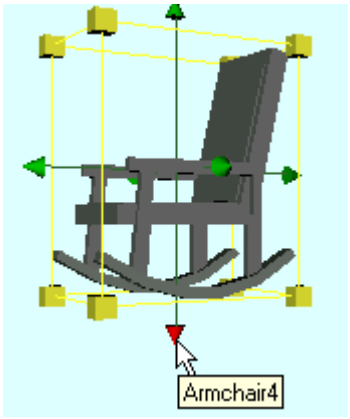
- Position the pointer over the object rotation control in the Perspective View window.
- Drag the object rotation control while holding down the CTRL key.

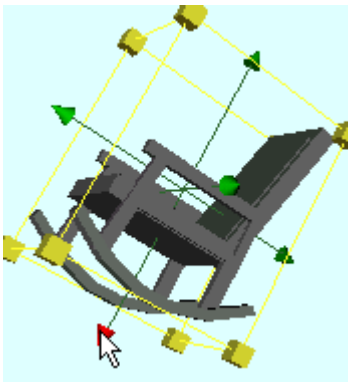
To set the object rotation controls in the center of the object bounding box:

- Double-click the green cone of the object rotation control.

Rotate an Object Directly in the Perspective View Window

You can rotate the object in the Perspective View window using the rotation controls: green cones at the end of the axes crossing the object center. The minimal angular step equals 15 degrees. For example, the following images demonstrate a rotation about the horizontal axis:





To achieve smooth object rotation:

- Hold down the SHIFT key while dragging the rotation control.

Note: If you rotate the object around the axis, parallel to the bottom face of its bounding box, the object becomes unsnapped from the face of the other scene item. Use snap the object to any visible face to attach the object to the face again.

Scale an Object Directly in the Perspective View Window

You can resize the object directly in the Perspective View window using the object scaling controls: yellow boxes in each bounding box vertex. When you position the mouse pointer over the scale control, the box is colored in red and the object name appears on the screen.

The following rules are applied for the scaling operations:

- The object is resized (scaled) in all dimensions simultaneously when you drag the scaling control: **increased**, if you move the pointer out of the opposite diagonal vertex of the object bounding box; **decreased**, when you move the pointer closer to the opposite diagonal vertex of the object bounding box.
- The object is resized in a single direction when you SHIFT-drag the scaling control. The initial movement determines the scaling direction.
- When you drag the scaling control while holding down the CTRL key the object is resized in all dimensions and the yellow box opposite the box you drag does not move. It is sometimes useful for the furniture arrangement in the room. The initial movement determines the scaling direction.
- If you scale the object, it can interconnect with the floor or the wall. Use this technique only if you actually need to insert the object into the other scene items. Try scaling the objects in the Perspective View window by yourself and find your own intuitive understanding of the scaling procedures.

Move an Object Directly in the Perspective View Window

To move the object in the Perspective View window:

1. Select the object you want to move by clicking on it in the Perspective View window.
2. Position the pointer over any face of yellow bounding box of the selected object. Drag the object to position where you want to place it. The face of bounding box you choose defines the plane in which you can move the object in a 3D space.

You can snap the object to any visible face of the other item in the Perspective View window:

1. Select the object you want to snap by clicking it in the Perspective View window.

2. Position the pointer over any face of yellow bounding box of the selected object. Drag the object while holding down the CTRL key to the appropriate (probably horizontal) face in the scene. The object becomes attached to that face and slides along it while you move the mouse. Note that if an object is moved to a face of another object, it becomes its sub-object.

You can transfer an object to another place in a scene using Scene Tree window:

- Drag the object icon from the Scene Tree to the position where you want to move the object in the Perspective View window. When the object is placed on any face, it is placed as if the face is holding it up. For example, when an object such as a chair is dragged to the vertical face of a scene, the chair is placed with its legs directly in contact with this face, as if the face was the ground. If an object is placed into the vacant space of a scene, the center of its bounding box is located 10m from the camera. If an object is moved to a face of another object, it becomes its sub-object.

Note:

- An object cannot be moved to its own face or to a face of its sub-object by the above method.

- When an object is moved, all its sub-objects are also moved or copied, retaining their orientation relative to the parent object.

Replacing with the Object from the Object Gallery

To replace an object in the scene with the object from the Object gallery:

1. Drag the object using the right mouse button from the Object gallery on the object you want to replace in the Perspective View, choose **Replace Object**.

2. In the Replace Object dialog specify the method by which the new object will be inserted.

Manipulating Plan

Choosing a Reference Coordinate System

The Coordinate System group lets you specify the coordinate system used for a transformation (Move, Rotate, Scale). The reference coordinate system determines the orientation of the X, Y, and Z axes used by the transform. The selected coordinate system affects positioning, rotation, and scale transformations.

Global coordinate system

The Global coordinate system is related to the VRML world or the *ISA* model space as a whole. World space is the universal coordinate system that *ISA* uses to position objects in the scene. When you look at a plan, you can see the vertical and horizontal rulers that measure distances for projections based on the world coordinate system. World space is constant and immovable. In the world coordinate system seen from the front, the X axis runs in a positive direction to the right, the Y axis runs in a positive direction upward, and the Z axis runs in a positive direction towards you (for details, see the figure in the "Change a view "). When you have the object selected in the Plan window, the coordinate axes displayed run in the direction of the VRML world coordinate axes, but their origin is moved and located at the object's local center – the pivot point. All

transformations can be done only about these shown axes. The object's pivot point specifies the transform center and affects scale and rotation transformations but has no effect on position transforms.

Local coordinate system

The local coordinate system is the coordinate system that relates specifically to the selected object.

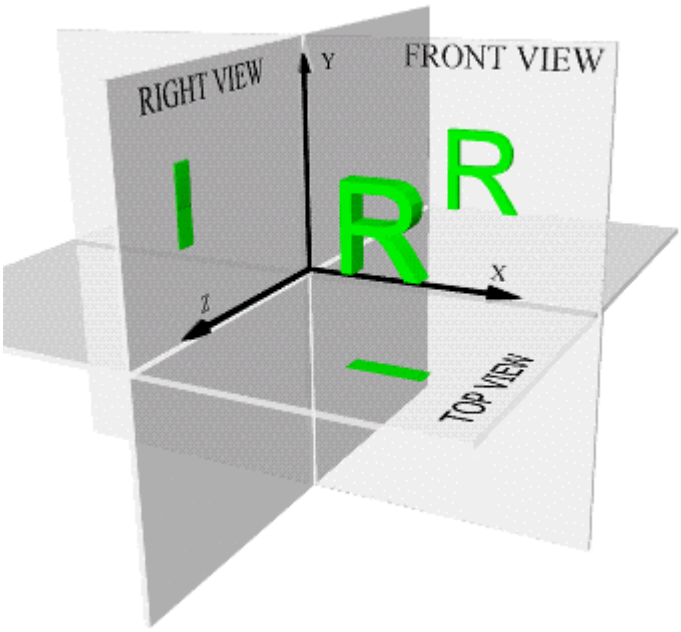
Each object has its own local center and coordinate system as defined by the location and orientation of the geometry elements forming the object. The local center and coordinate system of an object combine to define its object space. In contrast to the world coordinate system the direction of the object's X, Y, and Z axes depend on the current transforms of the object. You can see the difference between the two coordinate systems when you rotate an object around the object's local axis instead of the axes of the Global coordinate system. The axes would be rotated together with the object in this case.

Screen coordinate system

The two coordinate axes X and Y (Y is in the up direction) are in the plane of the computer screen. The Z axis runs towards you. The origin of this system is at the object's pivot point.

Change a View

Plan > Projection > Choose one of the Plan Projection command.



The Plan window represents six orthographic or right-angle projections based on the world coordinate system and the one for the user-defined plane. Objects on the plan are displayed as wireframes with no shading applied. Edges will be colored using the ObjectColor (blue by default) and SelectedObjectColor (red by default) attributes for the selected object (you can change the default colors in the Plan Properties dialog at

Colors tab). If the object is selected, the object pivot point and the coordinate axes, which correspond to the Coordinate System mode, will also be shown in plan.

To change a view:

- In the Plan Projection group on the Plan toolbar select one of the flyout buttons to choose among Top, Left, Front, Bottom, Rear, Right, and the User-defined projection correspondingly.

The user-defined plane

As you work, you can quickly change the view in plan and store it by specifying the user-defined plan projection. You can use either of two methods: to orient the plane manually – or align plan projection with the buttons from the group. For details on how to specify the user-defined plane by rotating a plan, see "Rotating a Plan".

If the local coordinate axes are shown in plan, in addition to the projection based on the world coordinate system, you can specify a projection plane which is based on the axes tripod of the object's local coordinate system. You can align the projection plane in such a way that its orientation would coincide with the plane that is defined by any two local axes. For example, if you choose Z and Y local axes, you'll get the right view of the object with its local axes aligned to horizontal and vertical lines of the computer screen. The upper button in the Align Plan Projection group activates the top view of object in plan.

To align a plan projection:

- Make sure you are in the Local Coordinate System mode, select the object, choose the corresponding view from the Align Plan Projection group on the Plan toolbar. The user-defined plane button is activated in the Plan Projection group and this indicates that the user-defined projection is on. *ISA* keeps this view until you select the other view from the Align Plan Projection group again or change the plane orientation by using Rotate Plan.

Move a Plan (Pan the View)

Plan > Manipulation Mode > Move Plan

You can move a plan by moving your mouse.

To move plan view:

1. Click Move Plan on the Plan toolbar, position the pointer anywhere in the Plan window, the pointer looks like a hand. Press the left mouse button.

2. Move the mouse while holding the left mouse button to move a plan.

Rotating a Plan

Plan menu > Manipulation mode > Rotate Plan;

Tools menu > Options > Plan > Plan Properties dialog.

You can change the orientation of the projection plane by rotating a plan. The rotation center can be different and depends on what the Tie Point center is selected under Modes in the Plan Properties dialog and where you have positioned the pointer.

To rotate a plan about the object's pivot point or center of bounding box:

1. Click the **Rotate Plan** button on the Plan toolbar.

2. Make sure the rotation center is set according to your preferences. To change it, right-click somewhere on the plan, and in the Plan Properties dialog under Modes choose the rotation center in the Tie Point list box.

3. Position the pointer on any line of the selected object, the pointer changes to . Press the left mouse button and move the mouse. The direction of the rotation (about the vertical or horizontal axis on a plan) is determined by your mouse movement.

Note:

- The Rotate Plan procedure changes the user-defined plan projection and activates the button immediately after the left mouse button is released.

- If you have positioned the pointer on an empty part of the plan the rotation center will be calculated as follows. Imagine a ray from the point where you have pressed the mouse button. It is perpendicular to the computer screen and runs away from you. The rotation center is the intersection between this ray and a plane which is parallel to the computer screen and contains the VRML coordinate origin.

Zooming a Plan

Plan > Manipulation Mode > choose Zoom Plan or Zoom By Rect.

To increase or decrease view magnification:

1. Click the Zoom Plan button on the Plan toolbar.

2. Position the pointer anywhere in the Plan window and press the left mouse button. The position where you have pressed the button determines the point on the plan about which the zooming will take place.

3. Move the mouse up or down to increase or decrease view magnification.

4. Release the left mouse button.

To magnify a selected part of a plan:

1. Click the Zoom by Rect button on the Plan toolbar.

2. Position the pointer anywhere in the Plan window.

3. Press the left mouse button and select a particular area in Plan by dragging the mouse.

4. Release the left mouse button. The selected area becomes fully visible in the plan.

Note:

- The grid shown spacing is changed as a result of the zooming.

Fit the Plan to an Object

Plan menu > Fit to Object.

To make the object view visible in the Plan window:

1. Select the object you want to make visible in the plan.
2. Click Fit to Object on the Plan toolbar.

Note:

- Objects on the plan appear in a wireframe, where the object and its sub-objects are included. The scale of the plan is adjusted so that all components of the object are visible.

Fit Plan to a Scene

To make a scene view fully visible in the Plan window:

- Click Fit to Scene on the Plan toolbar.

Note:

- The plan's scale is set so that all elements of the scene and optionally the camera Pinocchio Icon (this option is turned off by default), can be seen.

Using the Grid

Tools menu > Options > Plan. This activates the Plan Properties dialog box.

A grid is shown in the Plan window by default but it is activated automatically only in cases where any of the two axes of the selected coordinate system are aligned along the screen coordinate system. These appear as solid lines. Inactive grid displays appear as dotted lines and the grid snaps are impossible in this case. Choosing grid settings can simplify the construction process because the grid gives you a visual reference to distances in the scene and can be used as a reference system for using grid snaps. Note that grid spacing is the size of the grid's smallest square. If you zoom out a plan and the grid's smallest square becomes too small to be displayed the grid is enlarged.

To turn the grid on or off:

1. Choose Properties from the right-button pop-up menu in the Plan window. The Plan Properties dialog box appears. Click the General tab.
2. Select or clear the **Show** check box in the Grid box to turn the grid on or off.

To set grid spacing:

1. Choose Properties from the right-button pop-up menu in the Plan window. The Plan Properties dialog box appears. Click the General tab.

2. In the Grid box, adjust the value in the Spacing field, which is in current units.

To use a snap:

1. Choose Properties from the right-button pop-up menu in the Plan window. The Plan Properties dialog box appears. Click the General tab.

2. Under Grid, select the Snap check box to turn the snaps on.

Guide Lines

Tools menu > Options > Plan. This activates the Plan Properties dialog box.

ISA allows you to have an unlimited number of vertical and horizontal guide-lines in the Plan window. It's often useful to have a line to keep an eye on while you are assembling the scene in the Plan window. Guide-lines require activation before use. Make sure that the Show check box under Guide Lines at the General tab of the Plan Properties dialog box is selected (the default). To open the Plan Properties dialog box choose Properties from the right-button pop-up menu in the Plan window.

To include a guide line:

1. Position the pointer over the vertical or horizontal ruler in the Plan window and press the left mouse button.

2. Move the mouse to the plan area while holding down the left button. The Guide-line appears.

To move a guide line:

1. Position the pointer over the guide line to be moved. The pointer changes to a double arrow.

2. Drag the guide line.

To enter the precise value for a guide-line:

1. Make sure that one of the object manipulation buttons is pressed on the Plan toolbar (Move, Rotate, Scale, or Move Object Pivot).

2. Right-click the guide-line and choose Change Guide Line from the pop-up menu in the Plan window. The Change Guide Line dialog appears.

3. Specify the value for the new guide-line position in the Global coordinate system. Click OK.

Turn the Follow Selection On/Off

When you assemble a scene you have to find the object you need. By default *ISA* does not adjust the view in the Plan window to show the selected object. Setting the Follow Selection option on makes the projection of

the selected object visible. The coordinates of the plan is then adjusted according to the Tie Point parameter in the Modes group in the Plan Properties dialog (see “Setting Plan Window Options”). If the *Center of object bounding box* is selected the plan follows the center of the object’s bounding box. If *Object pivot* – the object transform center (see “Choosing a Transform Center”) is visible.

To turn the Follow Selection on or off:

- Right-click anywhere in the Plan window and choose Follow Selection.

Keyboard Commands

You can manipulate the Plan using keyboard commands. The following table shows the default keyboard shortcuts. Please note that before using these commands you should activate the Plan window by clicking anywhere on an empty part of the plan.

T – sets the Top view in plan.

B – sets the Bottom view in plan.

F – sets the Front view in plan.

K – sets the Back view in plan.

R – sets the Right view in plan.

L – sets the Left view in plan.

U – sets the user-defined projection.

Setting Plan Window Options

Tools menu > Options > Plan > Plan Properties dialog.

The Plan Properties dialog contains the options for the grid and guide-lines, placing the object in a scene when dropped in the plan, viewing mode, the object rotation center, units of measurements, the Fit To commands and color settings. You can open this dialog in two ways:

- Right-click anywhere in the plan and then choose Properties.
- Or on the **Tools** menu, click **Options**, and then click **Plan**.

The General tab

Grid group:

Show. Shows or hides the grid on plan.
Snap. Select to turn the grid snapping on.
Spacing. Specify the Grid cell in units, defined in the Units group.

Guide Lines group:

Show. Select this check box to turn the guide-lines on in the Plan window.
Snap. Select to turn the guide-line snapping on.

Units group:

When you change units, the program displays measurements in the new unit for your convenience. All dimensions in the Plan window read out in the new unit. No object is changed in this process. As in the physical world, objects in *ISA* maintain their absolute size, regardless of how you measure them.

Modes group:

Drop. This provides two different ways to place objects in the scene; to place an object on the top of an other underlying object in the scene; or position the object with its bottom face at zero level of the vertical axis (Y-axis in the VRML world).
Draw. Selecting *Hide invisible lines* hides edges which are formed by the invisible (beneath the faces on the top of plan) faces. If *All lines* is selected, all of the edges are visible in the wire frame view of plan.
Tie Point. This allows you to specify the transformation center for operations in plan, such as Rotate Plan and Follow Selection.

Fit Options group:

Fit to scene with camera. Modifies the Fit to Scene command (Plan menu). When selected, the command adjusts the plan's scale so that all elements of the scene, including the camera's Pinocchio icon, can be seen.
Fit to object with pivot. Modifies the Fit to Object command (Plan menu). When selected, the command adjusts the plan's scale so that the selected object, including its pivot point, can be seen.

The Colors tab

You can specify the colors for elements of the Plan window, such as grid, guide lines, the window background, and so on. Click the desired color from the palette or choose System Color from the list and then click OK.

Making a Scene Interactive

Overview of Fields, Events, and Routes

In *ISA*, the building block of the VRML scene is the object. If you are familiar with VRML97, *ISA's* object is represented by a prototype with the appropriate structure. These objects represent everything that can exist in the VRML world. Every object contains *fields* that hold the values of its parameters. For example, the directional light source (the DirLight object from the System gallery) has a *color* field that specifies the color properties of light emission. The basis for animation and interaction with a world is the ability to change the values of a given object's field. The way to change a field is to send an *event* to that field by means of a mechanism through which this event can be propagated to cause changes in the other object. This is called a *route*. The route is the connection between an object generating an event and an object receiving the event. You can represent the route as a path through which information passes from one object to another. For example, you can connect the Button object from the System gallery to the PointLight object, in order to turn on a light.

The *ISA's* object acts like a black box: the object's interface (the fields and events) provides the only contact possible between the inside and outside of the object. The object's interface includes the following elements: *field*, *exposedField*, *eventIn* and *eventOut*.

field
A data element contained in an object that define its attributes like name, state, size, direction and every value, is of a specific field type. If a field is not exposed, it has not associated events and you cannot route to or from it. A field can be given an initial value when you create a world but cannot be changed subsequently. All fields of the selected object are shown at the Properties tab of the Properties/Events window.

exposedField (exposed field)
This describes a field for which there are two associated events; an incoming event that lets you set the field's value; and the outgoing event that sends out the new value when the field's value changes. For an exposed field, you can wire a route to or from it. Because the exposed field has associated events, it is shown both in the Properties/Events and Routing Diagram windows.

eventIn (incoming event)
An object's input jack. The eventIn receives events when it is connected to a route and when an event is sent to it. All events are shown at Events tab of the Properties/Events window and in the Routing Diagram.

eventOut (outgoing event)
An object's output jack. The eventOut sends events out when it is connected to a route. All fields are shown at Events tab of the Properties/Events window and in the Routing Diagram.

abilities
ISA also gives the user a mechanism through which it is possible to assign to any object the additional features **–** *abilities*. This attribute is accessible at the Properties tab and facilitates the way to change and extend the functionality of the *ISA* object. For instance, the Touchable ability allows the user to interact with the object by clicking it and to generate the *TouchSensor.click* outgoing event as a result of the user click. If this sensor is "wired" to a light source, clicking on an object with the Touchable ability assigned could turn on a light.

Operating with the Properties of Objects

Objects' Properties

Any object from the Object gallery or VRML file located in a scene has at least the following attributes:

1. **Name –** the object name. Defines the name of the object in the Scene Tree. Objects appear in the Scene Tree under their gallery names (this can be changed).

2. **Owner** – indicates the parent of the object.

3. **Abilities** – indicate if the additional features are assigned to the object. These attribute also facilitate changing and extending the functionality of the *ISA* object:

   Touchable
   Collidable
   Hidable
   Movable
   Turnable
   Proximity
   Looksensitive
   Billboarded
   Highlighted
   Labeled
   Linked
   Camera-aligned
   DropObjectTarget
   DropTextureTarget

   See “Properties Reference Of Objects” for details on object abilities.

4. **Object specific attributes.** Groups of objects located in the Object gallery may have predefined properties that hold the values of the object parameters. For example, objects located in the System gallery support interaction with the user and provide light sources, sounds, the scene background, viewpoints and guided tours. Click the Help tab in the Properties/Events window to get an explanation of the predefined properties of objects. You can expose fields and events and therefore specify the object interface by using the Edit Object and Save Scene as Object commands.

Viewing Properties

To display object’s properties:

1. Select the object.

2. Click the Properties tab in the Properties/Events window. You will see the list of the properties for the selected object. It has two columns: the first column represents a name; and the second one indicates the value that defines the object’s property and may be changed.

Change Properties

To change the property of an object:

1. Select the object.

2. Click the Properties tab in the Properties/Events window.

3. Click anywhere in the value column to activate the property of an object that you are going to change. Some properties such as *Name, intensity, radius*, etc. may be changed manually using the keyboard. If the or button appears, click it and select the value from the drop-down list box, specify values in the appeared boxes, or use the Material Editor tool. The Add button allows you to specify an additional item in the case of multiple-valued fields; the Delete – removes the currently selected item. All changes are immediately reflected, either in the Perspective View or the Scene Tree window. Some properties have advanced editing interfaces which are described in later topics. For more details, see “Properties Editing Interfaces”.

Note:

Click the Help tab in the Properties/Events window to get an explanation of the properties of objects that have been shipped with *ISA*.

Properties Editing Interfaces

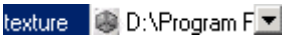
When the field is exposed via the object interface, it becomes visible at the Properties tab of the Properties/Events window. The type of field determines the interface for specifying and editing the corresponding property. See “Object Interface Control” for details on exposing fields from the *ISA* object.

The most useful interfaces:

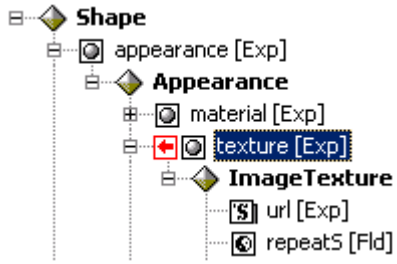
texture
material
SFString and MFString

texture

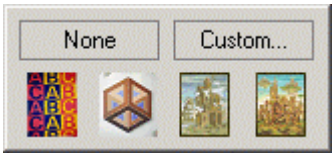
The *texture* field at the Properties tab:



The typical structure of an object with the texture exposed looks like the following:



Interface



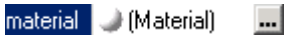
The *texture* field at the Properties tab allows you to look for and add a picture. Click anywhere in the *texture* field to activate the button with a down arrow and then click the button. If the folder specified at the *texture* field contains any image files, the folder is scanned and the dialog box displays small preview buttons, called thumbnails.

Procedures

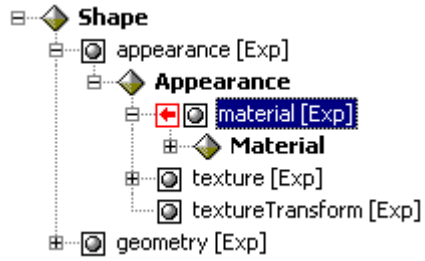
- To apply an image, click the picture thumbnail, or click Custom to activate the Open dialog box.
- To remove the image, click None. If there are no images applied, the symbol appears.
- To specify the path to the image manually, click the *texture* field at the Properties tab and type the path in the text area, or paste it with the Paste command from the right-mouse pop-up menu.

material

The *material* field at the Properties tab:



The typical structure of an object with the material exposed looks like the following:

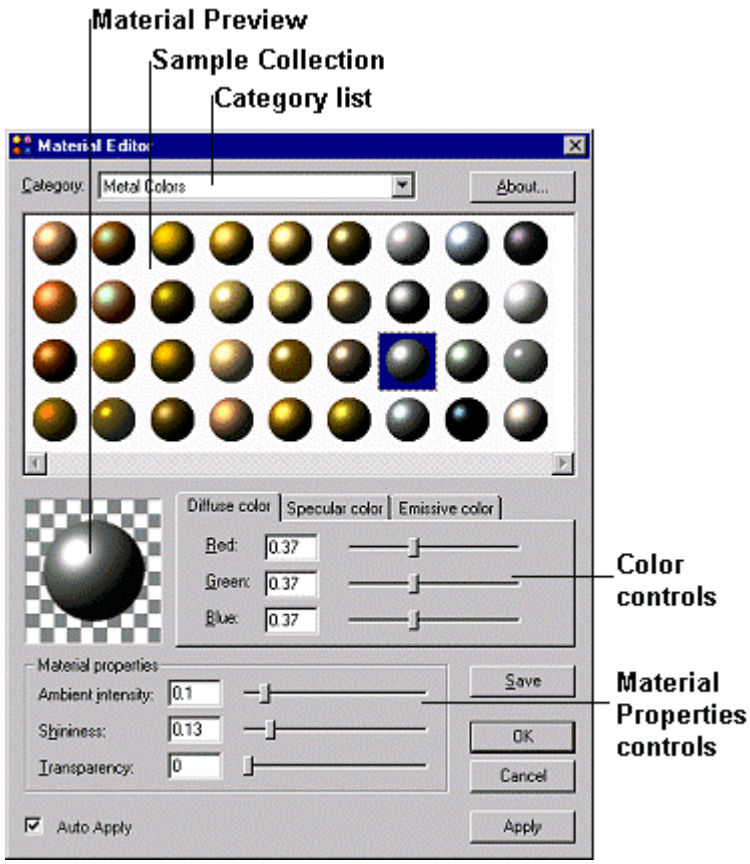


The Material Editor provides functions to assign material to individual shapes and to create and edit surface materials. Specifying a color and material properties allows you to add greater realism to a VRML scene. A material describes how an object reflects or transmits light.

To view the Material Editor

- click anywhere in the *material* field of the Properties/Events window and then the ... button that appears.

When you start the Material Editor you see an arrangement of screen elements, which is described below.



Material Preview.This displays the current material on a sample sphere. You can change the material by using the

Material Editor controls. To assign the material to the currently selected shapes, click Apply. To save the edited material to the User Defined Colors category, click Save.

Category list. The Category box lists the contents of a material library that you can use for selected shapes.

Sample Collection. Displays the collection of currently active material samples. You can select and preview the material on a sample sphere by clicking the material icon.

Color controls. These controls allow you to change a color component in the material.

- **Diffuse color** describes the color of light that an object reflects when illuminated by white light.
- **Specular color** determines the color of the specular area of a highly reflective object.
- **Emissive color** indicates that the object glows on its own.

Material Properties controls

These allow you to specify ambient, shininess, and transparency material components.

- **Shininess** determines the size of the specular area of a highly reflective object. Lower shininess values produce soft glows, while higher values result in sharper, smaller highlights.
- **Ambient intensity** specifies how much ambient light from light sources the surface will reflect. Ambient light in VRML simulates light that does not come directly from a light source. It is the general light that illuminates the entire scene. Ambient light is omni-directional and depends only on the number of light sources, not their positions with respect to the surface.
- **Transparency** specifies the degree of transparency of an object, ranging from 0.0 for completely opaque surface to 1.1 for completely "clear" surface.

Procedures

To apply a material to an object in a scene:

1. Make sure the *material* field is exposed in the selected object you want to assign the material to. You can, for example, expose the *material* field with the Edit Object command. Click the *material* field at the Properties tab.
2. Start Material Editor by clicking at the Properties tab.
3. You can modify a current material or specify a new one by using the Material Editor controls or by selecting a material from the material library. To select a readymade material, select the category of the material library that contains the material you want to assign, click the material icon in the Sample Collection window. The selected material appears in the Material Preview.
4. Click Apply to display changes in the Perspective View window. Click OK.

Note:

- To display changes in the scene immediately, select the Auto Apply check box.

To save the current material:

- Click the Save button. This saves the material into the User Defined Colors category library.

To delete the material from the User Defined Colors category:

- Select the material you want to delete by clicking its icon in the Sample Collection window, press Delete. There is no way to delete materials from other categories.

Note: When specifying a new material by using the Material Editor controls, you can press <Page Up>, <Page Down>, <Home>, <End>, and arrow keys to move the slider.

Designing Materials

A material's attributes describe the visual and optical properties of the material – its color, how it reflects light and how transparent it is. These are used by the VRML lighting equations during rendering. You can give an object the following appearance to determine how light reflects off an object to create color:

- **Diffuse color** describes the color of light that reflects most from the object. The surface with the diffuse component reflects all VRML light sources depending on the angle of the surface with respect to the light source. The more directly the surface faces the light, the more diffuse light is reflected.

- **Specular color** and **Shininess** determine the color and the size of the specular area on the highly reflective object, the color of the light that reflects directly back to the viewer. When the angle from the light to the surface is close to the angle from the surface to the viewer, the specular component is added to the diffuse and ambient color calculations. Lower shininess values produce soft glows, while higher values result in sharper, smaller highlights.

- **Emissive color** indicates that the object glows on its own. This can be useful, for instance, for displaying pre-lit models.

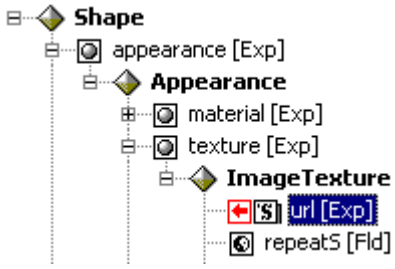
- **Ambient intensity** specifies how much ambient light from light sources the surface will reflect. Ambient light in VRML simulates light that does not come directly from a light source. It is the general light that illuminates the entire scene. Ambient light is omni-directional and depends only on the number of light sources, not their positions with respect to the surface.

- **Transparency** specifies the degree of transparency of an object, ranging from 0.0 for a completely opaque surface to 1.0 for a completely "clear" surface.

SFString and MFString

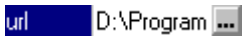
An SFString field contains a sequence of characters (a string). An MFString field contains zero or more string values. Many VRML nodes have fields of this type, as for example, Anchor, Background, ImageTexture, Inline, FontStyle, and others. In the case of the ImageTexture node the image is read from the URL specified by the *url* field. When you expose the MFString field, you can choose one of the two interfaces that appear on the Properties tab. This is controlled by setting the "Treat as File Path" option. For details, see “Object Interface Control”, Right-Click Menu (object interface).

The typical structure of an object with the *url* field exposed looks like the following:



Interface

The *url* field at the Properties tab with the "Treat as File Path" option turned on:

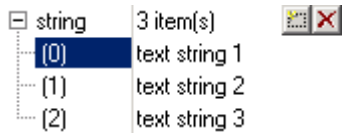


Procedures

- Click anywhere in the property field to activate it and then the ... button that appears to open a standard Open dialog or type a string value in the value column.

Interface

The SFString field at the Properties tab with the "Treat as File Path" option turned off:



Procedures

- To expand the string values, click the (+) symbol; to collapse, click (-).
- The Add button allows you to specify an additional item.
- The Delete button removes the currently selected item.
- Type anything you need in the text area or paste the string with the Paste command from the right-mouse pop-up menu.

Object Interaction

Interaction of Objects

Some objects generate events in response to environmental changes or user actions. An event is an indication that something has happened. The objects can be wired into a circuit by creating routes. You can represent the route as a path through which information passes from one object to another. For example, you can connect the Button object from the System gallery to the PointLight object in order to turn on a light.

You can create a route both in the Properties/Events and Routing Diagram windows. Once an object has generated an initial event, the event is propagated from the object producing the event along any routes to other objects. These other objects may respond by generating additional events, etc.

Events

An object receives incoming events and passes the events to another object. With events you can change the properties of an object or start animations.

To show object events:

1. Select the appropriate object.
2. Click the Events tab in the Properties/Events window.

You will see the list of events. Events are of two types; outgoing events send values; and incoming events receive values. The ◂ left-pointing arrow represents the incoming event and the ▸ right-pointing arrow represents the outgoing event. The following icons indicate the types of events:

Note:

- Objects located in the Object gallery may have additional predefined events. Click the Help tab in the Properties/Events window to get an explanation for the events of the selected object or see "Object Gallery" in the Appendixes of the document.

- Animator also adds events to the object such as: *[animation name].signal*, *[animation name].skipBack*, *[animation name].skipForward, [animation name].srart, [animation name].toggle*. Events to start an animation, for example *TouchSensor.click*, are added by the Trigger selected on the Animator toolbar. For more details on events and fields in Animator, see “Controlling Animation”.

- You can create new objects by using the *Save Scene as Object* command and therefore display events and fields of the objects forming a scene.

Routing in the Properties/Events Window

To create a route:

1. Select the object from which you want to create a route.

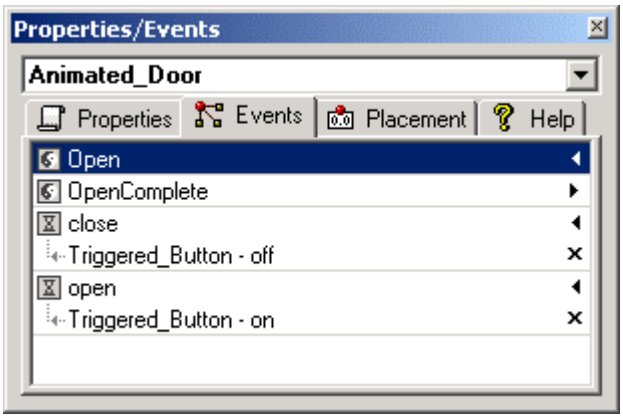
2. Click the Events tab in the Properties/Events window.

3. Click the outgoing event. The right-pointing arrow becomes red.

4. Select an object receiving the event. *ISA* will display only those incoming events to which you can create a route.

5. Click the incoming event.

For example, the following image demonstrates two routes from the Triggered Button to the Door object:



To remove a route:

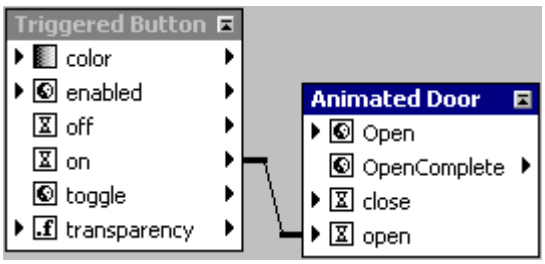
- Click the × cross.

Routing in the Routing Diagram Window

To create a route:

1. Click the Routing Diagram tab.
2. Position the pointer over the right-pointing arrow located at the right of the object (outgoing event).
3. Press the left mouse button and holding it down, stretch the route to the right-pointing arrow located at the left of the object (incoming event).

For example, the following image demonstrates a route from the Triggered Button to the Door object:



To remove a route:

1. Position the pointer over the route. It will be colored red (see Colors Preferences Settings for details on how to change the default color).
2. Right-click selected route, choose **Remove Route** from the pop-up menu.

You can also remove a route by double-clicking it.

To display information concerning routes:

- Right-click the route and choose **Route Info**. The appeared dialog box shows the names of connected objects, eventIn and eventOut in square brackets, and the type of event.

To jump to an object that sends or receives the event:

- Right-click the route and choose **Go to Source** or **Go to Target**.

To move or copy a route to another jack:

- Position the pointer over the end of the route you want to move, press the left mouse button and holding it down move the route to another location – or
- Drag the route with the CTRL key pressed to copy a route.

To select a route when several routes are joined together:

- Position the pointer over the end where routes meet and click until the route you want will be colored red. This is useful in cases when moving or copying routes to another jack.

Rules for Routing

- Incoming events are always routed to outgoing events and vice versa. You can't route two incoming events to each other, or two outgoing events to each other.
- The types of the events at both ends of the route must be identical. For example, a SFTime event to SFTime event.
- You can't route an outgoing event to an incoming event for the same field of the same object.

Forwarding Events to Console Window

You can view the event whenever it is generated in Preview mode by forwarding it to the Console screen. See “Console Window” for details on the interface.

To stop or cancel forwarding the events to Console:

- On the object’s icon in the Routing Diagram window, right-click the field or event whose output message you want to view in the Console screen, mark or clear **Output to Console**. If Output to Consol is on, the event name is shown in bold type.

Manipulating Objects in Routing Diagram

Move an Object in the Routing Diagram

You can change an object’s icon location in the Routing Diagram window for your convenience. To move an object, drag its title bar. If you move the object’s icon in such a way that the pointer is outside the Routing Diagram window, the icon will continue moving until you release the mouse button or position the pointer back to the window region. The direction in which you drag the mouse determines the object’s icon motion.

Depending on whether the *Move Routed Objects* option from right-button pop-up menu is turned on, the object will move together with objects connected to it.

Using the Place Selected command

1. Select the object whose icon you want to move in the Routing Diagram window.
2. Right-click the place in the Routing Diagram window where you want to have an icon and then click **Place Selected** in the pop-up menu.

Note:

To maximize or minimize an object’s icon, double-click the title bar or click the in the upper right corner of the object.

Hide or Show Objects

By default, *ISA* displays all objects in the Routing Diagram window that have exposed fields, eventIns or eventOuts. You can hide or show an object's icons in the Routing Diagram when routing in a scene. You can also hide or show objects with the *Hide/Show routed object* option turned on. This option is useful when you want to hide or show objects connected with each other. If you want to show or hide objects one by one, make sure the Hide/Show routed object is turned off.

To hide or show objects in the Routing Diagram window:

1. Right-click anywhere outside the object's icon in the Routing Diagram window, choose **Hide/Show Objects**. The Hide/Show Diagram Object dialog box appears.

2. Select or clear the check box corresponding to the object you want to show or hide.

   To hide or show all objects in the Routing Diagram window click **Hide All** or **Show All**. To hide objects without any routes in the Routing Diagram window, click **Hide Routeless**.

3. Click OK.

To quickly hide an object:

- Right-click the object's title bar, choose **Hide Object**.

To show the icon of the object that has been previously hidden:

- Select the object, right-click anywhere in a vacant place of the Routing Diagram window, and choose **Show Selected**.

To turn the Hide/Show routed object option On or Off:

- Right-click anywhere outside the object's icon and choose **Hide/Show Routed Object –** or

- Right-click anywhere outside the object's icon in the Routing Diagram window, choose **Hide/Show Objects**. The Hide/Show Diagram Object dialog box appears. Mark or clear the **Hide/Show routed object** check box below.

Find the Selected object

When you route the objects, you have to find the object you need. By default, *ISA* adjusts the view in the Routing Diagram window to show the selected object.

To turn the Follow Selection option On or Off:

- Right-click anywhere outside the object's icon in the Routing Diagram window and choose Follow Selection.

Turn the Snap Valid Routes On/Off

By default, the route automatically connects to the appropriate incoming event of the object when you drag the route in the vicinity of the object incoming event. This allows you to see whether the types of these events are identical and whether the events can be routed.

To turn the Snap Valid Routes On or Off:

- Right-click anywhere outside the object's icon in the Routing Diagram window and choose Snap Valid Routes.

Colors Preferences Settings

Right-click outside the object's icon in the Routing Diagram > choose Colors

Right-click the object's title bar > choose Border Color.

The following tasks are frequently performed when you want to choose the colors of the icons, borders, selections, background and routes in the Routing Diagram window for objects:

To change the color of the icon border together with the outgoing routes:

- Right-click the object's title bar, choose Border Color, and then adjust the appropriate color.

To change the window elements colors:

- Right-click anywhere outside the object's icon, choose Colors, click the color selector and choose the color you want from the palette. If you want to restore the colors to their default values, click Defaults in the Diagram Color Settings dialog. Click OK.

Advanced View

Right-click outside the object's icon in the Routing Diagram > choose Advanced View.

You can show or hide the additional transformation fields on the object's icon in the Routing Diagram window. If you are running more than one animation based on the object transformation, i.e. the rotation, scale and translation fields, turning this option on allows you to connect to the corresponding exposed field and in this way to transform the other object at the same time with the object for which you already have an animation.

Animating Objects

Animation

ISA lets you make a scene richer and more convincing by animating objects. You can change the position, orientation and size of any object in your scene over time. You can also change all its other properties, such as color, transparency, intensity, etc. as represented in the Properties/Events window. An object may have one or more animations associated with it. To create an animation, you should make a sequence of *keyframes* in the Animator window, and select the interpolation method. By default, *ISA* performs standard linear interpolation between keyframes. The additional incoming and outgoing events are associated with the object allowing you to control the animation playing. When the animation is created, the sensors on the Trigger list box let you define which events trigger the actions you specify in the Animator. The trigger also adds events to the object.

If the animated object has sub-objects, the movement animations of the object spread to all its sub-objects. For example, if a table is the object and a book on it its sub-object, the movement of the table is at the same time the movement of the book.

Each object in the scene has a parent. The movement animations of the object are related to location of the parent. Changing the parent's location, you also change the location of the point relative to which the object's movement animations are performed. If you move the object to another parent, its animation will be made relative to the coordinates of the new parent.

See also:

Animator Toolbar
Create an Animation
Preview an Animation
Controlling Animation
Selecting and Deselecting a Keyframe

Delete a Keyframe
Cutting and copying keyframes
Animating a viewpoint

Animator Toolbar

New Animation. Creates a new animation for the selection.

Delete Animation. Removes the animation you are currently editing. You can also delete the current animation by pressing the DEL key.

Displays the name of the animation that you are currently editing. You can select the other animation associated with the object from this drop-down list box. To change the name, click this box, type a new name and press ENTER.

This allows you to select a trigger from the drop-down list box. Lets you choose a sensor in the scene that will start your animation. Setting the trigger can assign the additional features at Abilities, events and routes to the selected object: World Entry, Touch Sensor, Proximity Sensor, Collision Group, Visibility Sensor, Custom Events (see Making a Scene Interactive, Controlling Animation for details).

Constant Interpolation. The value remains fixed until the next keyframe. No interpolation is performed.

Linear Interpolation. The value changes linearly from the current value to the next keyframe value.

Hermite Interpolation. The value passes through the key but it follows a smooth curve to avoid discontinuity.

Reverse Keyframes. Recalculates the keyframes to play current animation in an inverse manner.

Cut Keyframe. Cuts the selection and moves it to the *ISA* Clipboard.

Copy Keyframe. Copies the selection to the *ISA* Clipboard.

Paste Keyframe. Pastes the keyframe from the *ISA* Clipboard to the location of the time slider. If there are multiple keyframes, this places the first keyframe at the time slider location and all the others at the greater values on the timeline with the corresponding time intervals.

Record All Changes. When the Record All Changes button is red, you can click it to record all object changes.

Toggle the Stop Milestone. Lets you stop the animation at selected points on the timeline. If you click the button again, the stop mark is removed. By default, the animation always loops.

Toggle the Event Milestone. Generates the Animation.signal outgoing event at selected points on the timeline. With this event you can start the other animation.

Rewind. Sets the time slider at the beginning of the timeline.

Stop. Stops the animation.

Play. Lets you to preview the animation.

Forward. Sets the time slider at the end of the animation.

2.00 Displays the position of the time slider. You can also specify its position in this field.

Zoom In. Increases the view magnification of the timeline.

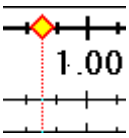
Zoom Default. Returns the view of the timeline to its original size.

Zoom Out. Reduces the view magnification of the timeline.

Create an Animation

To animate an object:

1. Click the Animator tab.
2. Select the object you want to animate.
3. Click the New Animation button on the Animator window toolbar.
4. On the left of the timeline, there is a list of the object's properties that can be changed in the animation process. Any object possesses the position property. An object may have other properties (these are the exposedFields of the object's interface), such as *color, transparency, intensity, textureTransform* and so on. Position property includes *translation, rotation, scale, orientation* and *center*. Click **Position** to open these properties. Click the buttons near properties to mark what you want to change in the current keyframe.
5. Drag the time slider (the yellow diamond) to a time other than 0.



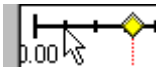
6. Move the selected object in your scene to the new position you want it to associate with at the time you set in step 5 or specify a new value for the exposedField at the Properties tab of the Properties/Events window. This association is called a keyframe. Note that the buttons near the properties you have changed in the current keyframe turn red.
7. Click the Record All Changes button on the Animator toolbar or manually select the fields that you are going to animate in the left bar. The first keyframe at 0 is recorded automatically.
8. Repeat steps 5-7 until you have set up all the keyframes in your animation.
9. To preview the animation, click the Play button on the Animator window toolbar.

Note:

You can also control the length of each cycle of already created animation by typing the cycle length, in seconds, in the *CycleTime* field at the Properties tab of the Properties/Events window.

Preview an Animation

To see how your animation will look, do one of the following:

- Click the Play button on the Animator toolbar to play the animation. To stop the animation at any time, click the Stop button.
- Drag the time slider back and forth to preview the animation.
- 

Click anywhere in the timeline to jump the time slider to that point.

Controlling Animation

When you create the animation, the additional incoming and outgoing events are associated with the object allowing you to control playing the animation. These are as follows: *signal*, *skipBack*, *skipForward*, *start*, and *toggle*. The name that appears in the list contains the animation name followed by the corresponding event name. For example, Animation1.signal.

[animation name].signal generates the SFTime outgoing event when the time slider meets the Toggle the Event Milestone green flag on the timeline. With this event you can start the other animation.

[animation name].skipBack the SFTime eventIn. Sets the animation at the beginning.

[animation name].skipForward the SFTime eventIn. Sets the animation to the last keyframe.

[animation name].start the SFTime eventIn. Starts animation playback.

[animation name].toggle if TRUE, plays the animation; if FALSE, stops.

The other events and fields can be added to the object depending on the trigger which is selected from the Trigger drop-down list on the Animator toolbar. This lets you choose a sensor in the scene that will start your animation, or in case of the Custom Events trigger, organize is so you define it yourself by sending events to the corresponding jack. The following are possible triggers:

World Entry. Starts the animation when the world is loaded into a browser. This kind of trigger takes a lot of processing power and will slow down your world. Adds the auxiliary WorldEntry object to a scene and connects *entry* of the WorldEntry with *[animation name].start*.

Touch Sensor. Starts the animation when the user clicks the specified object. Adds TouchSensor and connects *TouchSensor.click* with *[animation name].start*.

Proximity Sensor. Starts the animation when the camera enters a box-shaped region around the specified object. Adds ProximitySensor and connects Proximity*Sensor.click* with *[animation name].start*.

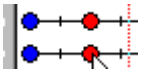
Collision Group. Starts the animation when the camera runs into an object. Adds Collider and connects *Collider.hit* with *[animation name].start*.

Visibility Sensor. Makes sure that the animation only runs when someone is looking at the animated object. Adds VisibilitySensor and connects *VisibilitySensor.enter* with *[animation name].start*.

Custom Events. Lets you choose a sensor in the scene that will start your animation manually by creating of routes. The Custom Events trigger does not add any events to the object in the scene.

Selecting and Deselecting a Keyframe

To select a keyframe:

-  To select a keyframe, click its blue dot. The selected keyframe turns red.
- To select multiple keyframes, CTRL-click additional keyframes to add them to the selection.
- To select all keyframes, right-click on an empty part of the Animator window and choose Select All.

Note:

Once you've selected keyframes, you can drag them around in time. You can also copy keyframes if you drag them while holding down the CTRL key.

To deselect a keyframe:

- To deselect a single keyframe, CTRL-click the keyframe.
- To deselect all keyframes, click on an empty part of the Animator window.

Delete a Keyframe

To remove a keyframe:

- Right-click the keyframes you want to delete, choose Delete.

Cutting, Copying and Pasting Keyframes

You can cut or copy keyframes onto the Clipboard and paste it on the timeline of the selected animation.

To cut or copy the keyframe(s) onto the Clipboard:

1. Select the keyframe(s) you want in the Animator window.
2. Click the Cut button on the Animator toolbar or choose Cut from the right-click pop-up menu – or

   Click the Copy button on the Animator toolbar or choose Copy from the right-click pop-up menu. The selected keyframe(s) is copied onto the Clipboard.

To paste the keyframe(s) from the Clipboard, do one of the following:

- Drag the time slider to a time where you want to copy the keyframe, or the first keyframe in the group, click the Paste button on the Animator toolbar.
- Right-click the place on the timeline where you want to paste the keyframe, or the first keyframe in the group. The pop-up menu appears. Choose Paste Here.

Instead of using the Clipboard, you can choose another ways to copy keyframes:

- Select the keyframe(s) you want to copy, then drag them while holding down the CTRL key.

Or

- Drag the time slider to the keyframe you want to copy. Hold down the CTRL key, then drag the time slider to a new position. Click the Record All button on the Animator toolbar to automatically record all object changes, or manually select fields you are going to animate in the left bar.

Animating a Viewpoint

You can animate the position and orientation of a viewpoint using the Animator in order to move the viewer around in the world. Use this feature to define the exact path the user follows through the scene. The process of animating looks much like the one for the geometrical object but requires some specific actions.

To animate a viewpoint:

1. Drag Viewpoint from the System gallery to an empty place in the Scene Tree.
2. To store the start camera position, make sure the Viewpoint is selected, then click the Store field at the Properties tab of the Properties/Events window, and then the button that appears.
3. Click the Animator tab and then the New Animation button on the Animator toolbar.
4. Select the Custom Events trigger from the

Trigger drop-down list box. This allows you to start your animation manually by creating routes. The other possible trigger is World Entry.
5. Select the interpolation method by clicking the appropriate button on the Animator toolbar.
6. Set the AutoStore to TRUE at the Properties tab of the Properties/Events window.
7. Move the camera to the second location by using navigation's commands in the Perspective View or by dragging the Pinocchio icon in Plan.
8. Drag the time slider (the yellow diamond) to the desired position.
9. Click the red record button on the Animator window toolbar to record the second keyframe. The first keyframe is recorded automatically.
10. Set up all the other keyframes in your animation by repeating steps 7-9.
11. Click rewind.
12. To avoid re-writing the first position of animated viewpoint, set the AutoStore to FALSE at the Properties tab of the Properties/Events window.
13. Create a route to activate the animated viewpoint in a scene. If you want the animation of the viewpoint to start every time, the viewpoint must become active. Do this by creating a route from **IsBound** to **Animation.toggle** field.

Note:

- You do not need to click the Store button at the Properties tab every time you want to set up the next keyframe. If *AutoStore* is set to TRUE, this automatically stores the current camera position and activates the Record All Changes button. If FALSE, use *Store* to store a camera position.

- After you've created or edited the animated viewpoint, click rewind to set the animation at the beginning and set the *AutoStore* to FALSE at the Properties tab.

- The change of FieldOfView at the Properties tab does not affect the field of view immediately. To examine changes, make sure the Viewpoint is selected, click the *GoTo* field at the Properties tab of the Properties/Events window and then the button that appears.

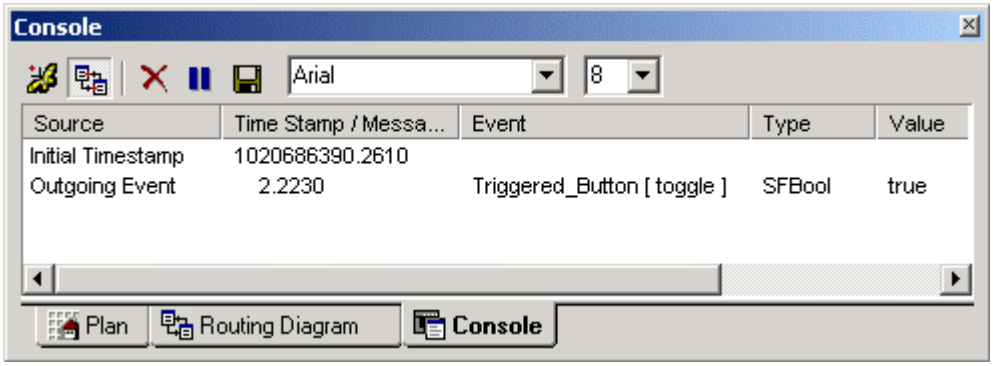
- While recording a keyframe, you can directly select which of the viewpoint's parameters should be entered: translation, rotation – or both. Click **Position** in Animator to open these properties. Click the radio buttons located close to properties in order to mark the change you want in the current keyframe.

Debugging

Console Window

ISA provides the Console window which allows you to view events during a scene preview. To open the Console window, click the Console tab located in the lower-left from the Perspective View window. See "Forwarding Events to Console Window" for details on displaying events on the Console screen.

Interface



Cortona Console Output. Shows or hides messages containing errors, warnings, and commentary information from the VRML console of Cortona VRML client. ParallelGraphics Cortona VRML client is used as a viewer in the Perspective View window.

Outgoing Events Information. Shows or hides information for outgoing events that have been marked in the Routing Diagram window by using the right-button pop-up menu.

Clear Console. Clears the events buffer.

Pause. Stops data output to the Console window but continues to output data to the events buffer. When you release the Pause button, all the most recent events will be displayed in the Console.

Log To File. Opens the standard dialog to specify a log file to be written.

Font box. Changes the font of the text or numbers in the Console window.

Font Size box. Changes the size of the text or numbers in the Console.

Columns:

Source. Indicates the source for the message. Cortona Console or Outgoing Event that is marked in the Routing Diagram.

Time Stamp / Messages The timestamp describes the time the event occurred. Messages are data strings sent by Cortona VRML client.

Event Indicates the object from which the event has been sent. The name of the outgoing event appears in square brackets.

Type Indicates the type of the outgoing event.

Value Displays the value of the occurred event.

Creating an ISA Object

The objects shipped with *ISA* have been designed mainly to make your scene interactive and therefore do not have any geometry to build models you need. *ISA* allows you to create your own galleries of 3D objects in VRML97 format. You can also use VRML 1.0 objects and scenes in *ISA* if you have Cortona VRML 1.0 Converter installed under Cortona VRML client (http://www.parallelgraphics.com/products/cortona/converter/).

You can use the following methods to create VRML models:

- Take any appropriate 3D modeling software tool, for instance 3D Studio Max, in which you build a model for further exporting to the VRML97 file format.
- Convert the existing CAD models to the VRML97 file format. It is recommended that you use ParallelGraphics Internet Model Optimizer (http://www.parallelgraphics.com/products/imo) to simplify CAD 3D models and reduce their file size.
- Use special VRML authoring tools – including the following: ParallelGraphics Internet Space Builder, VrmlPad, and Extrusion Editor. For more details, see http://www.parallelgraphics.com/products/

To convert a readymade VRML model to the file format of an *ISA* object:

1. Use the Open command to open the VRML file with the extension .wrl in *ISA.*
2. Use the Save Scene as Object command with the "Publish object" option turned on in order to save the new *ISA* object in the gallery.

To create an *ISA* object with animated sub-objects using the static VRML model:

1. Use the Import command to open the VRML file with the extension .wrl in *ISA*. When importing a VRML scene each Transform or Group node in it is converted to an *ISA* object.
2. Animate and add behavior to parts of a model. Use the Edit Object command to edit the interface of sub-objects.
3. Use the Save Scene as Object with the "Publish object" option turned on in order to save the new *ISA* object in the gallery.

To convert a part of model to an *ISA* object:

1. Use the Import command to open the VRML file with the extension .wrl in *ISA*. This converts any part of your model to an *ISA* object.
2. Use the Save Object command to save the only selected part of the model or the Save Selection command to save the selected part with its sub-objects, if there are any.
3. Use the Open command to open the object saved in the item 2.
4. Use the Save Scene as Object command with the "Publish object" option turned on in order to save the new *ISA* object in the gallery.

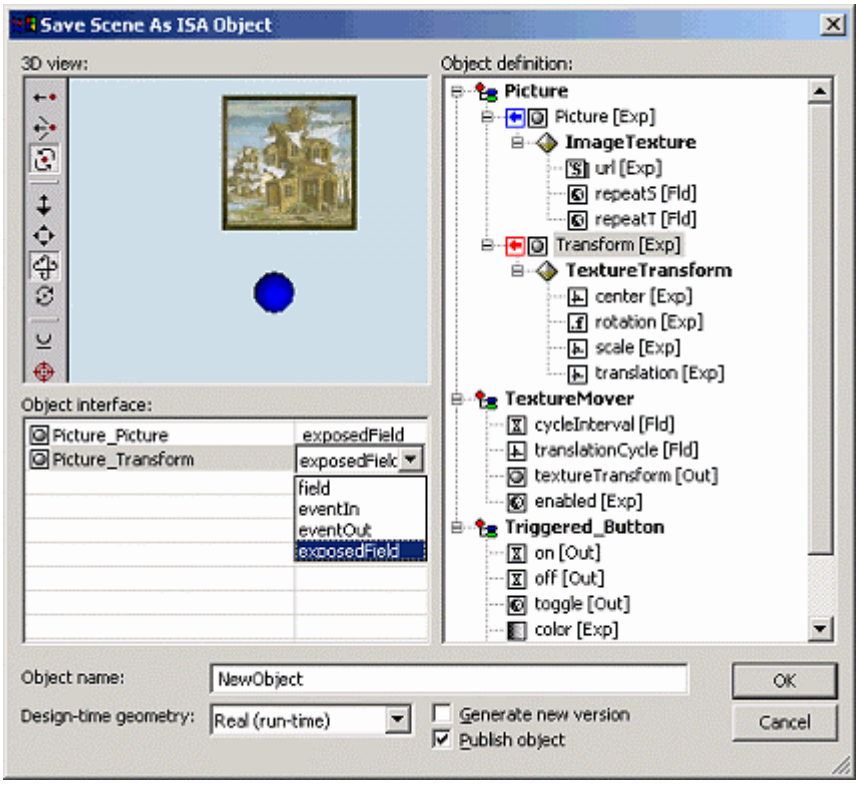
See also:

Editing Existing ISA Object
Creating ISA Objects Using ISB

Object Interface Control

The Edit Object or Save Scene As ISA Object dialog box appears when you choose the Edit Object or Save Scene as Object command (Object menu). These controls allow you to expose field and events so they are accessible in the selected object or scene that will be saved as a new *ISA* object.

Interface



3D view. Viewing and navigating the 3D scene or object to be saved or edited. You have a variety of navigation controls for viewing from the details of the smallest object to the complete extent of your scene. For details, see “Moving around: Walk, Fly, and Examine”. The click on the geometry shape in this window allows you to quickly find the corresponding branch in the hierarchy shown in Object definition. The selected *ISA* object or shape (a geometry node with nodes that define that geometry's appearance) becomes highlighted and the branch is expanded to display it.

Object definition. Represents the internal tree structure of the object or scene hierarchies. This displays the *ISA* objects and their interfaces – fields and events, the VRML nodes and fields and events that each node contains. A (+) symbol indicates that a level has collapsed. Click the (+) symbol to expand the level of the hierarchy.

This represents an *ISA* object in the VRML scene.

TR2 (Transform) The icon indicates the VRML node followed by its name under the DEF statement and the node name in brackets.

This icon marks the field or event added to the object interface.

The red icon represents the field or event selected in the "Object interface" list.

The name of the interface element appears in square brackets: [In] – eventIn, [Exp] – exposedField, [Out] – eventOut, [Fld] – field.

The following icons indicate the types of the events and fields:

Object interface. Lists the fields and events to be exposed in the object. Allows you to change the exposedField to another interface and share a set of commands available from the right-click menu for the item you choose.

A field name in the "Object interface" list cannot start with " 0", "1 ", "2 ", "3 ", "4 ", "5 ", "6 ", "7 ", "8 ", "9 ", " +", "-", "_" characters. The following characters shall not be used for field names: space, comma, point, separator character, comment, "#", "\", " {", "}", "[", "]". These characters are replaced with "?". Field names in the "Object interface" list must be different. Fields names are case sensitive.

Right-Click Menu (object interface)

Find Next Associated Field. If the item in the object interface refers to the several fields in the Object definition list, this commands selects the next associated field in the Object definition list. The currently selected field becomes grayed out.

Delete Current Association. If the item in the object interface refers to the several fields in the Object definition list, this command removes the currently selected association with the field in the object definition list.

Rename. This allows you to rename the item.

Delete. Removes the item from the "Object interface" list.

Treat as Percentage. When this option is selected the SFFloat exposedField will be measured in percents at the Properties tab of the Properties/Events window.

Treat as Angle. When this option is selected the SFFloat exposedField will be measured in degrees at the Properties tab of the Properties/Events window.

Treat as File Path. When this option is selected, the MFString exposedField at the Properties tab of the Properties/Events window provides the method to select the file. The button opens a standard Open dialog.

Object name. When you save a scene as a new *ISA* object, this specifies the object name.

Design-time geometry. Use this drop-down list box to change the appearance of the currently selected object. You can substitute a different geometry (ellipsoid, bounding box, translucent box) for your new object in the design mode, or even prevent rendering. You can accept the default "Real (run-time)" choice to display your object in the Perspective View or Plan windows as real geometry.

Generate new version. When the check box is on, the object version number is automatically increased.

Publish object. Use the Publish procedure to finalize your new object and possibly to use it on another computer. When the check box is on, the object will be published with the settings specified in the Publishing Option dialog. For more details about publishing, see “Setting Publishing Options”.

Procedures

To add the field or event to the object interface:

- In the Object definition list, double-click the field or event you want to be accessible in your new object. It appears in the "Object interface" list in the lower left – or
- Drag the field or event directly to the "Object interface" list.

Note: You cannot expose the SFNode or MFNode fields except *material*, *texture*, and *textureTransform* fields.

To create multiple association with the object interface:

- Drag the field or event on the existing item in the "Object interface" list. The types of the field or event at both windows must be identical.

To find the next association:

- Right-click the field in the "Object interface" and chose the Find Next Associated Field – or double-click the field.

To delete an item from the object interface:

- Right-click the item you want to delete, choose Delete – or
- Select the item by clicking then press Delete.

To change the name of the event or field:

- In the "Object interface" list, right-click the item you want to rename, choose Rename and type the new name and then press ENTER – or
- Click the item you want to rename, press F2, type the new name, and then press ENTER.

To change the exposedField to another interface element:

- Click the **exposedField** on the right side of the item, then open the drop-down list by pressing the button with a down arrow, and choose the new interface element. For details about fields and events, see “Overview of Fields, Events, and Routes”.

Editing existing ISA object

To edit an already existing *ISA* object:

1. Open an object. To open the object, you can drag the object from the gallery to *ISA’s* title bar.
2. Modify the scene, for example, by adding new objects and creating routes.
3. Choose Save Scene as Object from the Object menu. The Save Scene As ISA Object dialog box appears. All fields exposed in this object are displayed in the "Object interface" list.
4. You can add new exposed fields, change their names, and remove exposed fields that should not be used. For details, see “Save Scene as Object”.
5. Click OK then choose a gallery name where you want to save the object and an object file name.

Note:

When you mark the Generate New Version check box, the object version number is automatically increased.

Creating Internet Scene Assembler Objects Using Internet Space Builder

To create Internet Scene Assembler (*ISA*) objects using Internet Space Builder (*ISB*), you must follow one of the following procedures:

1. Create a 3D object by means of the rich tool kit from *ISB*. Set a uniform background using the Set Background command (File menu) to exclude the scene background from the new object.

2. Publish a scene applying the Publish command (File menu). We recommend that you publish a scene in a new folder to avoid the mixing of textures in different objects.

3. Open the published file with the extension .wrl in *ISA* and then use the Save Scene as Object with the "Publish object" option turned on in order to save the scene in the *ISA* gallery.

Or

1. Select an object in the Perspective View window of ISB.

2. Save an object in the Object gallery of ISB using the Save command (Object menu).

3. Open the saved file with the extension .vcd in *ISA* and then use the Save Scene as Object with the "Publish object" option turned on in order to save the scene in the *ISA* gallery.

Note:

This object will not have inherent scene nodes like Background, Viewpoint and DirectionalLight.

Using AVI Recorder

Key Features of AVI Recorder

The AVI Recorder allows you to create live video demonstrations of scenes created with *ISA* by recording the actions performed in the Perspective View window of *ISA*. You can save the recording as a movie that you can then play or send to others to play. The video file may be compressed by some common video codecs that are included with Windows.

Hardware Requirements:

- Random Access Memory (RAM): minimum of 64 MB.
- The video files you create with the AVI Recorder require space on your hard disk. The exact amount of space required will vary with the kind and amount of activity, such as recording time, size of recording area (Perspective View window) and screen resolution.

Record a Movie

1. Open a scene that you want to capture for a recording.
2. Click ▶ Play to enter previewing.
3. To start recording a movie, click R Record AVI on the View toolbar or from the View menu.
4. To stop recording, click ■ Stop, or click the R Record button again.

Note:

If the R Record button is depressed, the recording always proceeds in the Preview mode.

AVI Recorder Options

Tools menu > Options > AVI Recorder > AVI Recorder dialog.

To open the AVI Recorder dialog box, point to Options from the Tools menu, click AVI Recorder. This dialog box allows you to define the general parameters of the AVI Recorder.

Enable. Starts the recording of a movie in the Preview mode.

Use Compression. Ensures video file compression when you've stopped recording and before saving the video file.

OnLine Compression. Allows online video file compression for every frame. Do not use OnLine Compression on slow computers.

Current compressor. Displays the currently installed compressor.

Change... Allows you to select a video codec and modify its settings.

Used File. Displays the file name for your movie.

To look for a file.

Use this file for all records. Saves a movie with the file name displayed in the Used File box .

Enumerate file name. Adds a number that is consequently added for every record to the file name displayed in the Used File box.

Ask file for every record. Provides a dialog box for you to tape the name of every record.

Applies and saves any changes you have made and then closes this dialog box.

The cancel button closes this dialog box without saving any changes you have made.

Properties Reference Of Objects

Touchable – TouchSensor

Setting Touchable allows you to detect the mouse click or the pointer over the geometry of the selected object. As you position the pointer over the object with the Touchable ability assigned, the pointer changes to in Preview mode or Cortona VRML client. With events from this sensor you can change the properties of any object or start animations. This allows you to manipulate objects within the VRML world.

Adds the following events:

TouchSensor.click

indicates whether the mouse button is currently pressed. Generates SFTime event.

TouchSensor.enable

indicates whether the sensor is currently paying attention to pointer device. Receives SFBool event. If False, the sensor is turned off.

TouchSensor.over

indicates whether the mouse pointer is over the object. Generates SFTime event.

TouchSensor.toggle

indicates whether the mouse button is currently pressed. Generates TRUE on the first click, FALSE on the second and etc.

Adds the following object property:

Enabled

indicates whether the sensor is activated or not whenever you load the scene with the VRML browser. If False, the sensor is turned off.

Hidable – Cloak

Setting Collidable allows you to hide or show the object.

Adds the following object event:

Cloak.show

indicates whether the user can see an object. Receives SFBool event. If False, the object is not currently in view.

Adds the following object property:

Visible

indicates whether the sensor is activated or not whenever you load the scene with the VRML browser. If False, the object is not in view.

Collidable – Collider

Setting Collidable allows you to set up collision detection for a selected object. It determines whether the viewer has collided with an object.

Adds the following object's events:

Collider.enable

indicates whether collision detection is turned on or off. Receives SFBool Collider.enable event.

Collider.hit

generates SFTime outgoing event when the viewer collides with an object.

Adds the following object property:

Enabled

indicates whether collision detection is activated or not whenever you load the scene with the VRML browser. If False, the collision detection is turned off.

Movable – Mover

Setting Movable allows you to transform the pointer motion when moving an object. As you position the pointer over the object, the pointer changes to in Preview mode or Cortona VRML client, and you can drag the object while holding down the left mouse button in the plane that is defined by the *plane* property or *Mover.setPlane* event. Mouse movement with the left button pressed generates translation outputs that feels to the user as if the object is sliding along the selected plane.

Adds the following events:

Mover.enable

indicates whether the sensor is currently paying attention to pointer motion. Receives SFBool event.

Mover.isActive

when the user presses the left mouse button indicating the object's geometry, a Boolean TRUE event is sent.

Mover.setPlane

sending the SFInt32 event to this jack allows you to choose one of the planes in the local coordinates of the object: "0" defines YZ, "1" defines ZX, and "2" –- XY.

Mover.translation_changed

generates SFVec3F outgoing events whenever the user moves the object. It generates translation outputs specifying the point in the plane that the user is pointing to at any given moment during a drag.

Adds the following object properties:

Enabled

indicates whether or not the sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

Alternate

allows for moving the object in the horizontal or vertical plane (in the local coordinate system). If False, it is set for movement in the horizontal plane.

MinPos

constrains the translation. Indicates the minimal values along each axis.

MaxPos

constrains the translation. Indicates the maximum values along each axis. **Note:** If the *X* component of *MinPos* is greater then the *X* component of *MaxPos*, or the minimum *Y* is greater then the maximum *Y*, the translation is not constrained in that direction.

Plane

allows you to define the plane for movement in the local coordinate system of an object. Click anywhere in the field and choose the plane from the list.

Turnable – Rotator

Setting Turnable allows you to transform the mouse motion into a rotation of the object around its three axes.

As you position the pointer over the object, the pointer changes to in Preview mode or Cortona VRML client and you can rotate the object about its three axes while holding down the left mouse button.

Adds the following object's event:

Rotator.enable

indicates whether the sensor is currently paying attention to the pointer motion. Receives SFBool event.

Adds the following object properties:

Enabled

indicates whether or not the sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

MinAngle

constrains the rotation. Indicates the smallest value that the event from the mouse movement can contain. Any smaller value is changed to MinAngle.

MaxAngle

constrains the translation. Indicates the largest value that the event from the mouse movement can contain. Any larger value is changed to MinAngle.

Axis

allows you to define the axis for rotation. Click anywhere in the field and choose the coordinate axis from the list.

Proximity – ProximitySensor

Setting Proximity allows for generating events whenever the user moves into, out of, or within a box-shaped region. The region is shown with red bounding box in the Perspective View window if the scene is not previewed.

Adds the following events:

ProximySensor.enable

indicates whether the sensor is currently paying attention to user motion. Receives SFBool event.

ProximySensor.enter

indicates whether the user has entered the region. Generates SFTime event.

ProximySensor.leave

indicates whether the user has exited the region. Generates SFTime event.

ProximySensor.toggle

indicates whether the user has entered the region (TRUE) or has left the region (FALSE). Generates SFBool event.

Adds the following object properties:

Enabled

indicates whether or not the sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

Center

the center of the region in the local coordinate system, where the sensor detects the user motion. X, Y and Z are in units of the corresponding half-dimension of the object bounding box.

Size

the extent along each axis of the region around the defined *Center*. X, Y and Z are in units of the corresponding half dimension of the object bounding box.

Looksensitive – VisibilitySensor

Setting Looksensitive allows you to detect when the user can see an object. When the object is visible it can activate a behavior or animation. When the object is not in view it can deactivate the behavior or animation to improve the performance.

Adds the following events:

VisibilitySensor.enable

specifies whether the sensor is active. Receives SFBool event. If TRUE, the object detects changes to the visibility status of the object bounding box and sends outgoing events. If FALSE, the object does not send outgoing events.

VisibilitySensor.enter

indicates whether any portion of the object bounding box is in view. Generates SFTime event.

VisibilitySensor.leave

indicates whether the object bounding box is not in view. Generates SFTime event.

VisibilitySensor.toggle

indicates whether the object bounding box comes into view (TRUE) or remains out of view (FALSE). Generates SFBool event.

Adds the following object property:

Enabled

indicates whether or not the sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

Billboarded – Billboard

Setting Billboard allows you to include 2D images in your scene and to rotate them so that they are always facing the camera.

Adds the following object property:

ScreenAligned

specifies the axes used to perform the rotation. These axes are defined in the local coordinates of an object. If False, the vertical axis of an object is used for the rotation. True – both the vertical and horizontal axes are used.

Highlighted – DirLight

Setting Highlighted allows you to highlight the current object and all sub-objects.

Adds the following object's event:

DirLight.on

receives SFBool event. If TRUE, the object is highlighted. Repeats the received event as an outgoing event of the same type.

Adds the following object properties:

On

indicates whether the light is turned On (True) or Off (False) whenever you load the scene with the VRML browser.

Color

allows you to specify the color of the light emission.

Intensity

allows you to specify the brightness in the range from 0% (no light emission) to 100% (full intensity).

Labeled – Label

Allows the drawing of one or more text strings above the current object.

Adds the following object's events:

Label.Color

receives SFColor event and then repeats it as an outgoing event. Allows you to change the color of the text.

Label.Text

receives MFString event and then repeats it as an outgoing event. Allows you to change text strings.

Label.enable

allows you to show or hide the text. Receives SFBool event. If TRUE, the text is shown. If FALSE, the text is hidden.

Adds the following object properties:

Enabled

indicates whether or not the text string is in view when you load the scene with the VRML browser.

Text

allows you to specify any number of strings to be displayed above the object. Click the text field in the Properties window to activate the Add button and click it to enter the text.

Color

allows you to change the color of the text.

Transp

defines the degree of transparency of text in the range from 0% (completely opaque surface) to 100% (clear surface).

FontSize

allows you to change the height of the text.

Linked – Anchor

Allows you to link an Internet shortcut to an object.

Adds the following object properties:

Address

allows you to specify links or references to other files or scenes. Click the Address field and enter the desired address for a Web page or scene.

Params

provides the method with which to enter additional information for the VRML or HTML browser. This information is a string in the form "keyword=value". For example, some browsers allow the specification of a 'target' for a link to display a link in another part of an HTML document. The parameter field is then:

target=name_of_frame

Comment

allows you to enter a note on the current link.

Camera-aligned – Pin

This attribute allows you to move the object together with the viewer in such a way that the object remains static when using the navigation commands in the 3D window of the VRML browser. It is useful in cases where you want to show some elements controlling the scene, such as dashboard controls, are always in view.

Adds the following object properties:

position

defines the position without rotation of the object in the viewer's coordinate system (the system where the viewer is centered and looking down the Z-axis). It recommended that you find the required position by using the *Store* property and testing the result in the Preview mode or directly in the VRML browser.

Store

automatically stores the current object position in the *position* property. The viewer's coordinate system is assumed. To store the object position, move the object in Perspective View or Plan to the desired location, click anywhere in the Store field, and then click the appeared button.

DropObjectTarget – DropObjectSensor

This attribute assigned to the object allows you to drag and drop the link to the VRML model directly on the face of any object in the Cortona 3D window. The object associated with this link attaches with its center to the face where the mouse button has been released. It is useful in cases where you want to have an opportunity to add geometry objects to the scene already displayed by the Cortona VRML client, for example, furnishing virtual 3D rooms. Please note, that this ability uses ParallelGraphics VRML extension – the DropSensor node, and therefore requires Cortona for this functionality.

Adds the following object's events:

DropObjectSensor.enable

indicates whether the sensor is currently paying attention to pointer device. Receives SFBool event. If False, the sensor is turned off.

DropObjectSensor.objectDropped

generates the SFTime event when the object is dropped.

Adds the following object properties:

Enabled

indicates whether or not the DropObjectSensor sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

DropTextureTarget – DropTextureSensor

This attribute assigned to the object allows you to drag and drop the link to the texture on the face of any object in the Cortona 3D window. The texture associated with this link maps to the object's shapes. It is useful in cases where you want to have an opportunity to modify the appearance of geometry objects to the scene already displayed by the Cortona VRML client, for example, furnishing virtual 3D rooms. Please note, that this ability uses ParallelGraphics VRML extension – the DropSensor node and therefore requires Cortona for this functionality.

Adds the following object's events:

DropTextureSensor.dropTime

generates the SFTime event when the texture is dropped.

DropTextureSensor.enable

indicates whether the sensor is currently paying attention to pointer device. Receives SFBool event. If False, the sensor is turned off.

DropTextureSensor.texture

allows you to change an image visible on an object. Generates an outgoing SFNode event and specifies the references to the file containing a picture. You can use this event to change the texture or picture properties of the appropriate object.

Adds the following object properties:

Enabled

indicates whether or not the DropTextureSensor sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

Appendixes

Troubleshooting

This section contains answers to some of the most common questions asked by *ISA* users.

The unpredictable behavior of a character in a scene.

If a character does not move make sure the names of the objects that you chose as the checkpoints are mentioned in the checkpoints field of PathFinder at the Properties tab of the Properties/Events window.

If the chosen checkpoint objects are replaced you should also change the names for these new objects in the checkpoints field of PathFinder at the Properties tab of the Properties/Events window.

Live Camera does not follow the chosen object in a scene.

If the object chosen as a target is replaced you should also change the name of the object in the watchObject field of LiveCamera at the Properties tab of the Properties/Events window.

Why can I sometimes not animate the Viewpoint object? The Animator does not react to the moving camera.

It is strongly recommended that you activate the Properties tab of the Properties/Events window before you select the Viewpoint object. However, you may activate any other tabs in the Properties/Events window when the Viewpoint object is already selected. If the Viewpoint object is selected but Animator does not react to the camera moving, click the Properties tab in the Properties/Events window to correct this problem.

ISA Limits

Several restrictions exist regarding the size of scenes, objects, and the scale of a plan.

Maximum size of a scene: 100000m (100km).

Minimum size of a shape or object: 0.002m (2mm).

Minimum scale of a plan: 0.0001 m/pixel (0.1mm/pixel).

Maximum scale of a plan: 1m/pixel.

Object Gallery

CHARACTERS

Use objects from this gallery for including moving characters in your scene. You can define the trajectory by setting the checkpoints in order to move the character from one checkpoint to another. Incoming and outgoing events of all characters are analogous but the animations and their names can be different. See “Example with Characters” for details on how to make the character walk.

A Walking Character

The models from the Characters gallery have at least two readymade animations: Walk and Wait. This allows you to add a walking character to a scene. To make the character walk, you should add the character (Boy), PathFinder, and CheckPoint objects to the scene and then define signposts by adding names to the *checkpoint* field of the PathFinder object to interpolate the trajectory. You always have to connect the *destination_points* and *point_reached* fields of the PathFinder and the character (Boy) object. You also have to forward the *StartOut* outgoing event from PathFinder to character's animation (*ani_walk*).

Events:

ani_stopAll stops the current animation.

ani_wait starts the Wait animation. The character moves slightly to imitate breathing.

ani_walk starts the Walk animation.

animation_finished generates a signal when the animation stops.

animation_started generates a signal when the animation starts.

destination_point receives the information as to where the character should move. This should be connected with the destination_point field of the PathFinder object.

point_reached generates a signal that the character has reached the signpost. Should be connected with the point_reached field of the PathFinder object.

set_rotation allows for the rotation of the character.

set_translation allows for the translation of the character.

PathFinder

Adding the PathFinder object allows you to run the Walk animation and to forward the character towards checkpoints in the scene.

Properties:

checkpoints provides the method to choose the checkpoint in the scene. Specify the objects' names to define the character's trajectory in order to make the character walk from one checkpoint to another.

loop the character moves/does not move indefinitely between checkpoints when it is TRUE or FALSE.

Events:

StartIn receiving the SFTime event activates the PathFinder to look for checkpoints defined in the checkpoints properties field. You have to forward the event to this field to make the character walk along the trajectory specified at *checkpoints*.

StartOut lets you run the character's animation. Repeats the StartIn incoming event as the outgoing one.

StopIn receiving the event informs the PathFinder not to look for the next checkpoint.

StopOut **r**epeats the StartIn incoming event as the outgoing one and generates an outgoing event when the character has reached the last checkpoint and the loop field has been set to FALSE.

destination_point generates an outgoing event to forward the character to the next checkpoint in the scene. You should connect this field with the appropriate (the same name) field of the character object.

point_reached receives the information that the character has reached the current checkpoint. You should connect this field with the appropriate (the same name) field of the character object.

CheckPoint

Adding these objects to scene enables you to define the character's trajectory in order to make the character walk from one checkpoint to another. **Note:** the CheckPoint object is not visible in the Preview mode or VRML browser.

Properties:

color provides the method for choosing the color of the CheckPoint object from the palette.

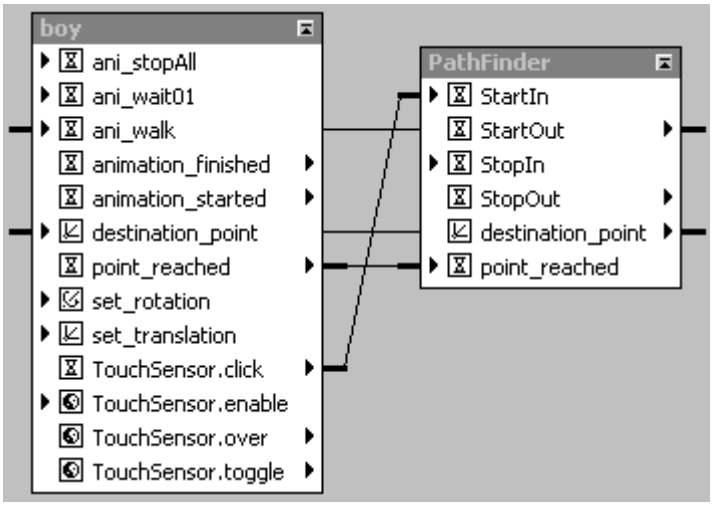
transparency defines the degree of transparency of the CheckPoint object (a cylinder) in a range from 0% (completely opaque surface) to 100% (clear surface).

Example with Characters

The following example demonstrates how to include the walking character in your scene:

1. Make sure you exit the Preview mode.
2. Drag Checkpoint, Boy from the Characters gallery and Chair from Basic gallery into the Perspective View window as shown in the figure:

3. Drag PathFinder from the Characters gallery into the Perspective View window.
4. Select Boy in the Scene Tree window. Click the abilities field in the Properties/Events window and then the arrow button which appears. Choose Touchable from the drop-down dialog box.
5. Click the Routing Diagram tab.
6. Connect objects in the Routing Diagram window as shown in the figure:



Note: You have to always connect the **destination_point** and **point_reached** fields of the PathFinder and the character object. You also have to forward the **StartOut** outgoing event from PathFinder to the character's animation.

7. Select Pathfinder in the Scene Tree window. Click the checkpoints field in the Properties/Events window and then the Add button and the arrow button which appear. Choose Chair from the drop-down dialog box. We've now created the first signpost.

   Note: The checkpoint objects should be at the same hierarchical level as the character object.

8. Click the Add button and then the arrow button. Choose CheckPoint from the drop-down dialog box. We've now created the second signpost. Note that CheckPoint is invisible in the Preview mode or in a VRML browser.

9. Click the loop field and choose True from the drop-down dialog box to make the character walk indefinitely between signposts.

10. To Examine the scene, click the character in the Preview mode.

Note: When the character moves to the checkpoint object, its target point is the center of the object (the object manipulator's moving point locates close to the center of an object). For example, to make the character walk in the horizontal plane, make sure that all centers of the checkpoint objects are in the same plane. It is also recommended that you use the Checkpoint objects from the Characters gallery as checkpoints.

CONTROL

This gallery contains objects that are useful for developing 3D demonstrations and manuals.

Delay

Use this object to delay the SFTime event.

Properties:

delay specifies the delay, in seconds.

Events:

timeIn the incoming event of the SFTime type that should be delayed.

timeOut the outgoing event of the SFTime type is generated after *timeIn* with the time interval specified in *delay*.

toggle the outgoing event of the SFBool type is generated with the value TRUE when the *timeIn* occurs and with the value FALSE at the moment of the event generated on the *timeOut* jack.

DelayBool

Use this object to delay the SFBool event.

Properties:

delay specifies the delay, in seconds.

Events:

boolIn the incoming event of the SFBool type that should be delayed.

boolOut the outgoing event of the SFBool type is generated after *boolIn* with the time interval specified in *delay*.

toggle the outgoing event of the SFBool type is generated with the value TRUE when the *BoolIn* occurs and with the value FALSE at the moment of the event generated on the *BoolOut* jack.

DemoControl

It is useful in cases when the developer needs a control panel in the scene. In the viewing mode the control panel is positioned at a fixed place in the VRML browser window and does not depend on the viewer position. The panel contains four control buttons to select each step of the demonstration: Previous, Pause, Play, Next, and also the field to display the number of the current step.

Properties:

Visible allows you to specify initial state of the menu. If TRUE, the menu is visible. If FALSE – hidden.

position specifies the relative shift in the VRML browser window about the default position (at the bottom).

NumberOfSteps specifies the number of steps.

ShowCurStepNumber If TRUE, the control panel will display the current step.

AutoNext specifies whether the automatic activation of the next step is On or Off. If TRUE, when receiving the event at *enable*, the current step number is increased by 1 without pressing the NEXT button.

AutoPlay specifies whether the automatic playing of all steps is activated (TRUE) or not (FALSE). If TRUE, when receiving the event at *enable*, the current step number is increased by 1 without choosing PLAY and the object generates the *stepPlay* outgoing event.

PanelTransparency defines the degree of transparency of the panel in a range from 0% (completely opaque surface) to 100% (clear surface).

The following properties allow you to set up colors for panels and the text:

colorB_on the border of the button when the pointer is over it.

colorB_off the border of the button.

colorM_on background of the selected button.

colorM_off background of the button

colorT_on the button symbol when the pointer is over it.

colorT_off the button symbol.

colorT_sel the symbol of the Pause button when pressed.

colorT_dis the symbol of a currently unavailable button.

Events:

show shows (TRUE) or hides (FALSE) the panel when receiving the corresponding event of the SFBool type.

autoPlay turns On (TRUE) or Off (FALSE) the automatic activation of the next step.

enable receiving the event of the SFTime type makes the button of the panel accessible for the user. The event should be routed with eventOuts that sends a signal about the end of the current step.

stepPlay the outgoing event of the SFInt32 type with the value that equals the current step number. The numbering begins with zero. The event is generated when the user clicks PLAY.

stepChanged the outgoing event of the SFInt32 type with the value of the current step number. The numbering begins with zero. The event is generated whenever the user clicks NEXT or PREVIOUSE.

stepPause the outgoing event of the SFInt32 type with the value of the current step. The numbering begins with zero. The event is generated whenever the user clicks PAUSE.

stepResume the outgoing event of the SFInt32 type with the value of the current step. The numbering begins with zero. The event is generated whenever the user clicks PAUSE for the second time.

play the outgoing event of the SFTime type. The event is generated whenever the user clicks PLAY. Simultaneously, all the buttons excepting the PAUSE button become inaccessible until they receive the *enable* event.

pause the outgoing event of the SFTime type. The event is generated whenever the user clicks PAUSE.

resume the outgoing event of the SFTime type. The event is generated whenever the user clicks PAUSE for the second time.

reset the incoming event of the SFTime type sets the panel to its initial state.

EventCounter

The object counts the number of incoming events of the SFTime type to *event_in*. It also can count from the beginning when the number of counts reaches *Module*, i.e. the counter has a scale of *Module*.

Properties:

CountByModule allows you to turn on or off the Count By Module mode. By default, this mode is not activated (FALSE).

Module specifies an integer to set up a scale of the counter. The default value is 2.

Events:

event_in an object's input jack for events of the SFTime to count.

reset receiving the incoming event of the SFTime sets the counter to "0".

out1_skipped the outgoing event of the SFTime. It is generated when receiving event_in when the number does not equal the Module value. If *Module* equals 2, this event is generated for odd received events on the *event_in* jack.

out2_accepted the outgoing event of the SFTime. It is generated when receiving *event_in* when number is equal the Module value. If *Module* equals 2, this event is generated for even events on the *event_in* jack.

out_bool_accepted the outgoing event of the SFBool type is generated when receiving an event on the *event_in* jack. The value of the event is set to TRUE if the number of the *event_in* equals *Module* and FALSE for all others.

eventNumber the outgoing event of the SFInt32 type – the number of events that have been received by the *event_in* jack. The event is generated every time when the object receiving an incoming event on the *event_in* jack.

Eye_to

This adds a semi-transparent sphere with a color that can be changed and a readymade animation to attract the user attention. In the editing mode this object is displayed as a yellow bounding box.

Properties:

Color defines the sphere color.

CycleTime defines the number of animations to be played.

Events

start the incoming event of the SFTime type starts the animation.

toggle the incoming event of the SFBool type starts (TRUE) or stops (FALSE) the animation.

stopped the outgoing event of the SFTime type is generated when the animation is finished.

JSfunction

This object allows you to call a Java Script function specified in the parent HTML page.

Properties:

functionName defines the name of the Java Script function

Events

call the incoming event of the SFTime type that initiates the function call.

callBool the incoming event of the SFBool type that initiates the function call with the parameter that equals the event value.

callStep the incoming event of the SFInt32 type that initiates the function call with the parameter that equals the event value.

Menu

This object is used to add two-dimensional text menu to a scene. The number of items and their values (the text strings) can be specified by setting object's properties. In the scene editing mode every menu item is the text line formatted according to the menu size.

Properties:

Visible allows you to specify the initial state of the menu when loading a scene. If TRUE, the menu is visible. If FALSE, it is hidden.

items provides the method to specify the number of lines and contents of the menu. Click the plus sign to display the list of lines. Click the text item at Properties tab to activate the Add and Delete buttons. You can type the word or phrase for the selected item. Use the UTF-8 encoding for national letters in the text field. You can use, for example, ParallelGraphics ASCII to UTF8 Converter.

PanelWidth specifies the text width, in metres. If the text in *items* exceeds the *PanelWidth,* the text will be automatically shrunk.

HighlightSel specifies if the selected by the user item will be highlighted. The default value is TRUE. The color to highlight is *colorT_sel*.

PanelTransparency defines the degree of transparency of the panel in a range from 0% (completely opaque surface) to 100% (clear surface).

The following properties allow you to set up colors for panels and text:

colorB_on the border color of the item when the pointer is over it.

colorB_off the border color of the item.

colorM_on the menu item color when the pointer is over it.

colorM_off the menu item color.

colorT_on the text color when the pointer is over it.

colorT_off the text color.

colorT_sel the color of the selected menu item if the *HighlightSel* is TRUE.

Events

show receiving the SFBool event displays (TRUE) or hides (FALSE) the menu.

itemSelected generates the outgoing event of the SFInt32 type with the value equaling the number of the selected menu item. This numbering begins with zero.

Slider

The Slider object creates a slider control. The control is a rule containing a slider. When the user moves the slider, using the mouse, the control sends notification messages to indicate the change.

Properties:

initialValue sets the starting position for a slider when it appears in Preview mode.

integerMode allows you to choose on of two modes: integer or float. If TRUE, the slider generates discrete values. If FALSE, it allows you to select a set of consecutive values in a range.

Material sets the material properties for the slider control. Click the Open button to start the Material Editor that provides functions to assign material.

maxValue sets the maximum position for a slider.

minValue sets the minimum position for a slider.

Events:

floatValue generates SFFloat events indicating the current position of the slider.

integerValue generates SFInt32 events indicating the current position of the slider.

StepControl

The StepControl object is used to start any activity in the scene by sending the only event to the *play* jack with the ability to temporarily stop playing, for example, in order to guide the user though the current step of a 3D demonstration. The step, in turn, contains several actions. Activation of the following actions occurs by sending one of the five outgoing events from the StepControl object. The next outgoing event from the *start[N+1]* jack is generated only when the object receives an incoming event at the *stop[N]* jack which is a signal when the action stops. The object is designed to start five actions but combining the objects together it is possible to increase the number of actions to be played.

Properties:

delay defines the delay time, in seconds, from receiving the *play* incoming event to generating the *start1* outgoing event. This delay is necessary in cases where the preparatory commands are to be executed with the *playOut* event before the first action.

NumberOfStart defines the number of actions in a step to be played. This number is also used to generate *startNext* whenever the last action finishes or the number of finished actions equals 5.

Enabled allows you to define the initial state of an object when opening a scene. If TRUE, the objects runs. If FALSE, the object ignores the incoming events.

Events

enable receiving the event of the SFBool type changes the Enabled property.

pause receiving the event of the SFTime type stops generation of events until the object receives the event at *resume* jack. Enters the "PAUSE" mode.

pauseOut the outgoing event of the SFTime type is generated when the *pause* event occurs. This event can be used to transmit the event from one StepControl object to another.

play receiving the event of the SFTime type starts the events generating them one by one for the current step.

playOut the outgoing event of the SFTime type is generated when receiving the *play* event. This can be used for initiating preparatory actions in a scene, for example, changing the viewer position, setting animation at the beginning, etc.

resume receiving the event of the SFTime type exits the object from the "PAUSE" mode. The object generates actions one by one.

resumeOut the outgoing event of the SFTime type is generated when the resume event occurs. This event can be used to transmit the event from one StepControl object to another.

start1 it is generated after the *play* event occurs with the *delay* specified.

start2 it is generated when *stop1* occurs.

start3 it is generated when *stop2* occurs.

start4 it is generated when *stop3* occurs.

start5 it is generated when *stop4* occurs.

startNext it is generated when either the *stop[NumberOfStarts]* receives the event (the last action finish), or the number of finished actions equals 5. This event marks that this step of the demonstration is over and can be routed with the *enable* jack of the *DemoControl* object or with the *play* jack of the other instance of the StepContol object.

The following are the input jacks of the SFTime type that should be routed with the eventOuts giving a signal about the end of current action of the demonstration: **stop1**, **stop2**, **stop3**, **stop4**, **stop5**.

Note:

There must be a delay between the generation of *start[N]* and receiving an event at *stop[N]* jack. In the case of the instant execution of the action, you should use the Delay object to delay an event.

TimerToggle

This object is used to generate the sequence of Boolean events (TRUE, FALSE, …, etc.) with the time interval specified.

Properties:

ToggleLoop defines the time period for the timer to work from its initialization by the *start* event, in seconds.

ToggleInterval defines the time interval between the outgoing events, in seconds.

Events:

start receiving the SFTime event starts the timer.

stopped the SFTime event is generated when the timer stops.

toggle the SFBool event is generated with the interval specified with *ToggleInterval*.

IMAGE

The Image gallery contains objects that are useful for applying and modifying pictures or textures.

Movie

Allows you to add a movie to a scene.

Properties:

AutoStart if TRUE, starts playing after loading the scene in VRML browser.

loop if TRUE, the movie is repeated indefinitely.

speed specifies how fast the movie will be played. A *speed* of 2 indicates the movie plays twice as fast.

url provides the method to select the AVI or MPEG file to be played. Click the *url* field to activate the Open button, then click it or type the URL of a valid file.

Events:

isActive sends a Boolean TRUE event when a movie is playing.

start when receiving the SFTime event, starts playing a movie.

stop stops the movie.

Picture

Allows you to add a picture or photo to a scene.

Properties:

Picture allows you to look for and add a picture. Click anywhere in the Picture field to activate the arrow button. Click the arrow button. Click the picture you need or click Custom to activate the Open dialog box.

Transform This allows you to transform the picture. To change the values of the texture transformation, click anywhere in the Transform field to activate the arrow button. Click the arrow button and enter the new values.

Translation allows you to translate the picture. (0.5,0) means that the picture is translated to the left by half its width. (0,0.5) – downward by half its height.

Center specifies the center for the rotation and scale picture transformation. (0,0) means the lower-left corner; (-1,-1) – the upper-right corner of the picture. The default is (-0.5, -0.5), for the center of the picture.

Rotation specifies a picture rotation in degrees about the center – in a clockwise direction.

Scale specifies how to scale the picture on the face. (2,1) means that the picture is compressed by 2 and repeated twice in the horizontal. (1,2) means the same but in the vertical.

Events:

Picture allows you to change a picture visibly. When the incoming event is of the SFNode type and specifies the references to an other file containing a picture, the picture changes. Note: you can generate these events using the Animator and changing the texture or picture properties of the appropriate object. Repeats an incoming SFNode event as an outgoing one.

Transform when receiving the incoming event of the of the SFNode type, changes the current transformation of a picture. Repeats an incoming SFNode event as an outgoing one.

TextureMover

This object is used to create moving textures. You can animate them to simulate waterfalls, smoke, or moving clouds. The *textureTransform* outgoing event should be routed to any object that has the corresponding field of the *textureTransform* type and can display the picture. For example, the Picture object has the *Transform* input jack of this type.

Properties:

cycleInterval the time interval to move a picture by the part of its size that is specified in *textureTransform*, in seconds. Must be greater than zero.

enabled indicates whether the texture moving is activated or not in the scene when you load the scene with the VRML browser. If FALSE, the object will not generate outgoing events.

translationStep defines the vector to move the texture in X (horizontal) and Y (vertical) directions of the texture coordinate system. Because the texture coordinates, not the texture, are transformed, the moving direction appears reversed when viewed on the surface of geometry. (-1,-1) means that the picture is moved from the lower-left to the upper-right corner of the Picture object. This length of the vector also specifies the speed at which the texture may move – the more the values of X and Y, the faster the texture moving.

Events:

enabled indicates whether the TextureMover is activated or not in the scene when you load the scene with the VRML browser. If FALSE, the object will not generate outgoing events to move the picture.

transform if *enabled* is TRUE, generates SFNode events to animate a texture in the direction specified in *translationCycle*. This outgoing event should be routed to any object that has the corresponding field of the *textureTransform* type, for example, to the *Transform* input jack of the Picture object.

TextureSlideShow

The *TextureSlideShow* object generates SFNode outgoing events to show all images specified in the *texture* property. The *texture* outgoing event should be routed to any object that has the corresponding field of the *texture* type and can display the picture. For example, the Picture object has the *Picture* input jack of this type.

Properties:

cycleinterval the time interval to show a picture, in seconds. Must be greater than zero.

enabled indicates whether the slide show mode is activated or not in the scene when you load the scene with the VRML browser. If FALSE, the object will not generate outgoing events.

texture allows you to look for and choose pictures visible in an other object.

To add a picture to a list, click anywhere in the *texture* field of the Properties/Events window to activate the Add button, click the Add button to add a new item, click the down arrow button that appears and click Custom. The Open dialog box appears. Choose any image file for texture.

To delete the selected item, click .

Events:

enabled receiving the SFBool event activates (TRUE) or not (FALSE) the slide show.

texture if the object is activated, it generates the outgoing events of the SFNode type containing the picture to display.

LOGICAL

The Logical gallery contains objects to control the events.

AdapterBool

When boolIn receives the SFBool event, the object generates two outgoing events: *boolOut* of the SFBool type and *timeOut* of the SFTime. This object is usually used as a concentrator of routes of the SFBool events to combine several incoming events to the only outgoing event. It also can be used in unsophisticated cases instead of the Time Bool Converter.

Events:

boolIn the incoming SFBool event.

boolOut the outgoing SFBool event.

timeOut the outgoing SFTime event.

AdapterTime

When *timeIn* receives the SFTime event, the object generates two outgoing events; the SFTime *timeOut*, and the SFBool *boolOut*. This object is usually used as a concentrator of routes of SFTime events to combine several incoming events to the only outgoing event. It also can be used in unsophisticated cases instead of the Time Bool Converter.

Property:

boolValue specifies the value of the outgoing event. The default value is TRUE.

Events:

timeIn the incoming SFTime event

timeOut the outgoing SFTime event

boolOut the outgoing SFBool which value is set to boolValue.

Boolsorter

Repeats an incoming Boolean event as an outgoing one on one of two output jacks. When *inBool* receives TRUE, it is forwarded to *outTrue*; when FALSE, to *outFalse*.

Boolean Filter

Repeats an incoming Boolean event as an outgoing one if *state* is set to TRUE.

Properties:

state indicates whether *state* is set to TRUE or FALSE whenever you load the scene with the VRML browser. The default value is FALSE.

Events:

Possible incoming and outgoing events are summarized in the following table. F – Boolean value FALSE, T – TRUE, N – no outgoing event, M – outgoing event of SFTime type.

In	T F T F
state	T T F F
---	---
out	T F N N
error	N N M M

When the **set_state** field receives an event of Boolean type, the **state** changes to the value of the event.

When the **set_state_Off** field receives an event of SFTime type, the **state** changes to FALSE.

When the **set_state_On** field receives an event of SFTime type, the **state** changes to TRUE.

Time Filter

Repeats an incoming SFTime event as an outgoing one if the *state* is set to TRUE.

Properties:

state indicates whether *state* is set to TRUE or FALSE whenever you load the scene with the VRML browser. The default value is FALSE.

Events:

Possible incoming and outgoing events are summarized in the following table. F – Boolean value FALSE, T – TRUE, N – no outgoing event, M – incoming or outgoing event of SFTime type.

timeIn	M M
state	T F
---	---
timeOut	M N
error	N M

When the **set_state** field receives an event of Boolean type, the **state** changes to the value of the event.

When the **set_state_Off** field receives an event of SFTime type, the **state** changes to FALSE.

When the **set_state_On** field receives an event of SFTime type, the **state** changes to TRUE.

Inverter

The Inverter object converts the incoming Boolean event into an outgoing event as shown in the following table. F – Boolean value FALSE, T – TRUE.

InBool	T	F
outBool	F	T

Logical AND

When receiving an incoming event the Logical AND object generates TRUE if *state1* and *state2* are both set to TRUE, otherwise, it generates FALSE.

Properties:

State1

indicates whether State1 is set to TRUE or FALSE (the default value) when you load the scene with the VRML browser.

State2

indicates whether State2 is set to TRUE or FALSE (the default value) when you load the scene with the VRML browser.

Events:

When inBool1 or inBool2 receives an incoming event, the value of State1 or State2 changes to the value of the corresponding event.

The outBool outgoing event is generated every time an object receives an incoming event. Possible outgoing events of the Logical AND object are summarized in the following table. F – Boolean value FALSE, T – TRUE.

State1	T	T	F	F
State2	T	F	T	F
outBool	T	F	F	F

Logical OR

When receiving an incoming event the Logical OR object generates TRUE if either *state1* or *state2* is TRUE, otherwise, it generates FALSE.

Properties:

State1

indicates whether State1 is set to TRUE or FALSE (the default value) whenever you load the scene with the VRML browser.

State2

indicates whether State2 is set to TRUE or FALSE (the default value) whenever you load the scene with the VRML browser.

Events:

When inBool1 or inBool2 receives an incoming event, the value of State1 or State2 changes to the value of corresponding event.

The outBool outgoing event is generated every time an object receives an incoming event. Possible outgoing events of the Logical OR object are summarized in the following table. F – Boolean value FALSE, T – TRUE.

State1 T T F F
State2 T F T F
---
outBool T T T F

Logical XOR

When receiving an incoming event the Logical OR object generates TRUE if *state1* does not equal *state2*, otherwise it generates FALSE.

Properties:

State1

indicates whether State1 is set to TRUE or FALSE (the default value) whenever you load the scene with the VRML browser.

State2

indicates whether State2 is set to TRUE or FALSE (the default value) whenever you load the scene with the VRML browser.

Events:

When inBool1 or inBool2 receives an incoming event, the value of State1 or State2 changes to the value of corresponding event.

The outBool outgoing event is generated every time when an object receives an incoming event. Possible outgoing events of the Logical XOR object are summarized in the following table. F – Boolean value FALSE, T – TRUE.

State1 T T F F
State2 T F T F
---
outBool F T T F

Selector

When receiving the incoming event of the SFInt32 type by *nEvent* (the number of event), it generates one of the five SFTime outgoing events according to the *nEvent* number. If *nEvent* = 0, the outgoing event appears at *event_out1,* and so on.

If *nEvent*>4 it generates the *nEventDec* outgoing event of the SFInt32 type and its value is set to *nEvent-5.*

When routing from *nEventDec* to the incoming *nEvent* of the second instance of the Selector object, the combination of the two objects forms the new selector that operates with 10 incoming numbers. This way you can create the selector with an extended range of incoming numbers.

Events:

nEvent the incoming event of the SFInt 32 type that specifies the number of the object's output jack. This numbering begins with zero. When routing, this input jack is usually connected with the eventOut of the object that can interactively specify a number, for example, Menu and DemoControl, or EventCounter.

nEventDec the outgoing event of the SFInt32 type. If *nEvent* >4, it generates *nEventDec=nEvent-5.*

The following are outgoing events of the SFTime type: *event_out1, event_out2, event_out3, event_out4, event_out5.*

Selector Bool

When receiving the incoming event of the SFInt32 type by *nEvent* (the number of event), it generates five outgoing events of the SFBool type with values specified according to the *nEvent* number. The *event_out* with the number *nEvent-1 is set to* TRUE*.* All others are set to FALSE.

If *nEvent* >4, it generates the *nEventDec* outgoing event of the SFInt32 type and its value is set to *nEvent-5.*

When routing from *nEventDec* to incoming *nEvent* of the second instance of the Selector object, the combination of two objects forms the new selector that operates with 10 incoming numbers. This way you can create the selector with an extended range of incoming numbers.

Events:

nEvent the incoming event of the SFInt 32 type that specifies the values of the object's output jacks to TRUE or FALSE. The numbering begins with zero. When routing, the *nEvent* input jack is usually connected with the eventOut of the object that can interactively specify a number, for example, Menu, DemoControl, and EventCounter.

nEventDec the outgoing event of the SFInt32 type. If *nEvent*>4, it generates *nEventDec=nEvent-5*.

The following are outgoing events of the SFBool type: *event_out1, event_out2, event_out3, event_out4, event_out5.*

SwitchBool

When receiving an event of the SFBool type at *event_in,* it generates one of the five outgoing events of the SFBool type. The number of the *event_out* outgoing jack is specified by the *Choice* property. This numbering for *Choice* begins with zero, i.e. if *Choice=0*, the event is generated at the *event_out1* jack.

If *Choice* is out of the [0,4] range, there is no outgoing event.

The incoming event of the SFInt32 type at the *choice* jack changes the *Choice* property. Simultaneously, the SFInt32 *choiceDec* is generated with the value *choice –5.*

When routing from *choiceDec* to the incoming *choice* jack of the second instance of the SwitchBool object, the combination of two objects forms the new object that operates with 10 outgoing jacks. This way you can create the SwitchBool with an increased number of events for commutating.

Properties:

Choice this allows you to initially select the object's output jack for commutating.

Events:

choice the incoming event of the SFInt32 type that changes the Choice Property.

choiceDec the outgoing event of the SFInt32 with the value set *choice –5.* It is generated when *choice* receives the event.

event_in the object's input jack for the SFBool events.

The following are outgoing events of the SFBool type: *event_out1, event_out2, event_out3, event_out4, event_out5.*

SwitchNode

Use this object to redirect events with information about texture or picture applied, and their transformation.

When receiving an event of the SFNode type at *event_in,* it generates one of the five outgoing events of the SFNode type. The number of the *event_out* outgoing jack is specified by the *Choice* property. This numbering for *Choice* begins with zero, i.e. if *Choice=0*, the event is generated at the *event_out1* jack.

If *Choice* is out of the [0,4] range, there is no outgoing event.

The incoming event of the SFInt32 type at the *choice* jack changes the *Choice* property. Simultaneously, the SFInt32 *choiceDec* is generated with the value *choice –5.*

When routing from *choiceDec* to the incoming *choice* jack of the second instance of the SwitchTime object, the combination of two objects forms the new object that operates with 10 outgoing jacks. This way you can create the SwitchNode with an increased number of events for commutating.

Properties:

Choice this allows you to initially select the object's output jack for commutating.

Events:

choice the incoming event of the SFInt32 type that changes the Choice Property.

choiceDec the outgoing event of the SFInt32 with the value set *choice –5.* It is generated when *choice* receives the event.

event_in the object's input jack for the SFNode events.

The following are outgoing events of the SFNode type: *event_out1, event_out2, event_out3, event_out4 event_out5.*

SwitchTime

When receiving an event of the SFTime type at *event_in,* it generates one of the five outgoing events of the SFTime type. The number of the *event_out* outgoing jack is specified by the *Choice* property. This numbering for *Choice* begins with zero, i.e. if *Choice=0*, the event is generated at the *event_out1* jack.

If *Choice* is out of the [0,4] range, there is no outgoing event.

The incoming event of the SFInt32 type at the *choice* jack changes the *Choice* property. Simultaneously, the SFInt32 *choiceDec* is generated with the value *choice –5.*

When routing from *choiceDec* to the incoming *choice* jack of the second instance of the *SwitchTime* object, the combination of two objects forms the new object that operates with 10 outgoing jacks. This way you can create the *SwitchTime* with an increased number of events for commutating.

Properties:

Choice this allows you to initially select the object's output jack for commutating.

Events:

choice the incoming event of the SFInt32 type that changes the Choice Property.

choiceDec the outgoing event of the SFInt32 with the value set *choice –5.* It is generated when *choice* receives the event.

event_in the object's input jack for the SFTime events.

The following are outgoing events of the SFTime type: **event_out1**, **event_out2**, **event_out3**, **event_out4**, **event_out5**.

Time Bool Converter

The Time Bool Converter object converts a Boolean and SFTime incoming event into a SFTime or Boolean outgoing event.

When **inBool** receives TRUE, **outTimeOn** generates the SFTime event.
When **inBool** receives FALSE, **outTimeOff** generates the event of the SFTime type.
When **inTimeOn** receives the SFTime event, **outBool** generates TRUE.
When **inTimeOff** receives the SFTime event, **outBool** generates FALSE.

WorldEntry

The WorldEntry object allows you to generate an event when the world is loaded into a browser. This allows you to start your animation.

Event:

entry generates an outgoing SFTime event when the world is loaded into a VRML browser.

WorldEntryBool

The WorldEntry object allows you to generate an event when the world is loaded into a browser. This allows you to start your animation.

Event:

state generates an outgoing SFBool event when the world is loaded into a VRML browser.

NAVIGATOR

Live Camera

A Live Camera is a viewpoint that follows an object chosen in a scene. It is useful in cases where you want to show some object moving in the scene. A green cone represents the Live Camera object in the Perspective View and defines its initial position in a scene. Note that the Live Camera moves only in the horizontal plane.

Properties:

bindFirst if TRUE, it activates the viewpoint of Live Camera whenever you load the scene with the VRML browser.

description a viewpoint name which the browser will show.

jump controls a jump method from the previous viewpoint to the Live Camera viewpoint.
TRUE – instantaneous transition to the Live Camera viewpoint.
FALSE – a jump with the transition effect.

maxAngleSpeed the maximum angular speed at which the camera may rotate, in radian per second.

maxDistance the maximum distance at which the camera may leave the object, in metres.

maxSpeed the maximum speed with which the camera moves, in metres per second.

minDistance the closest distance which the camera may approach the object, in metres.

seeTranslation allows for a target point position change of an object which is used by the camera to detect the object motion. The target point is positioned at (X, Y, Z) in the local coordinate system of an object.

watchObject provides a method for you to search for an object which is followed by the camera. Click somewhere in the watchObject field and then select the object currently included in the scene from the list.

Events:

set_bind if TRUE, activates the viewpoint of the Live Camera. If FALSE, unbinds the viewpoint to change it back to the previous one.

set_bind_on when it receives an SFTime event, it activates the Live Camera viewpoint.

set_bind_off when it receives an SFTime event, it unbinds the viewpoint to change it back to the previous one.

Map

The Map object allows you to have a drawing of the scene's surface or a part of it as a visualization aid to see the current viewer position in the VRML world. The image of the map does not move relative to the computer screen when the user moves through a 3D space. In the Preview mode, or VRML browser, the user can move the image of map on the computer screen and change its scale by dragging the diamond on the lower-right corner of the picture.

When using the Map object, you should position the Map object in the "scene root" (the highest level of transformation hierarchy when the object has no hierarchical parent).

A drawing of the scene's surface, specified in *Picture*, must be represented as the top view of the VRML world (XZ-plane) where X (horizontal) corresponds to the S-axis of the texture map and -Z (vertical) – to the T-axis.

To maximize or minimize the map, click the image of map in the Preview mode or VRML browser.

Properties:

CursorMaterial sets material properties for the pointer on the map. Click the Open button to start the Material Editor that provides functions to assign material to individual shapes and to create and edit surface materials.

CursorScale changes the size of the pointer on the map.

MapScale changes the size of the map the Preview mode, or VRML browser.

Move_enabled indicates whether the map should be moved or scaled. If FALSE, the user cannot move and scale the map.

On enables you to hide (FALSE) the image of map in the Preview mode or VRML browser. The default value is TRUE.

Picture allows you to look for and specify a picture to be used for a map. Click anywhere in the Picture field to activate the arrow button. Click the arrow button. Click the picture you need or click Custom to activate the Open dialog box.

StartPos specifies the point of location of the map in the coordinate system which is moved together with the activate viewpoint. This defines the tree value to move the map in X (horizontal), Y (vertical), and Z (from the viewer) directions.

Events:

on when receiving an incoming event, displays the map if it was previously hidden.

off when receiving an incoming event, hides the map if it was previously shown.

Locator

The Locator object allows you to have a drawing of the scene's surface or a part of it as a visualization aid to see the current position of any object in the VRML world that has been selected on the scene design stage. The image of map added with this object does not move relative to the computer screen when the user moves through a 3D space.

When using the Locator object, you should position both the Locator and selected object in the "scene root" (the highest level of transformation hierarchy when the object has no hierarchical parent).

A drawing of the scene's surface, specified in Picture, must be represented as the top view of the VRML world (XZ-plane) where X (horizontal) corresponds to the S-axis of the texture map and -Z (vertical) - to the T-axis.

To maximize or minimize the map, click the image of map in the Preview mode or VRML browser.

Properties:

CursorMaterial sets material properties for the pointer on the map. Click the Open button to start the Material Editor that provides functions to assign material to individual shapes and to create and edit surface materials.

CursorScale changes the size of the pointer on the map.

ForwardDirection changes the direction which is used by the pointer to detect the object orientation. The default orientation of the pointer is parallel to the (0 0 -1) vector in the local coordinate system of an object.

MapScale changes the size of the map the Preview mode, or VRML browser.

Move_enabled indicates whether the map should be moved or scaled. If FALSE, the user cannot move and scale the map.

Object provides the method to choose the object in the scene. The position of the selected object will be displayed on the map.

On enables you to hide (FALSE) the image of map in the Preview mode or VRML browser. The default value is TRUE.

Picture allows you to look for and specify a picture to be used for a map. Click anywhere in the Picture field to activate the arrow button. Click the arrow button. Click the picture you need or click Custom to activate the Open dialog box.

StartPos specifies the point of location of the map in the coordinate system which is moved together with the activate viewpoint. This defines the tree value to move the map in X (horizontal), Y (vertical), and Z (from the viewer) directions.

Events:

Picture allows you to change a picture visibly. When the incoming event is of the SFNode type and specifies the references to other file containing a picture, the picture changes. Note: you can generate these events

using the Animator and changing the texture or picture properties of the appropriate object. Repeats an incoming SFNode event as an outgoing one.

on when receiving an incoming event, displays the map if it was previously hidden.

off when receiving an incoming event, hides the map if it was previously shown.

SYSTEM

The Logical gallery contains objects that basically correspond to the VRML nodes such as lights, Viewpoint, Fog, Group, Lod, NavigationInfo, ProximitySensor and others.

Background

Background allows you to specify a set of smoothly graded colors for both the ground and the sky in your scene. The sky forms a sphere around your scene. This sphere has a horizontal gradation of colors which ranges from SkyHoriz to SkyZenith. Similarly the ground forms a hemisphere under your scene with an infinite radius but drawn as though it is inside the sky sphere.

Properties:

Bind allows you to activate the current object in the *ISA* editing mode and therefore to reflect changes from it in the scene. When several Background objects are in the scene only one object may be active at a time. In the editing mode, *ISA* binds the first Background object encountered in the Scene Tree window. To activate an object, click anywhere in the *Bind* field of the Properties/Events and then the button that appears.

GroundHoriz the ground color closest to the horizon.

GroundNadir the Nadir (the bottom) ground color.

SkyHoriz the sky color closest to the horizon.

SkyZenith the Zenith sky color.

UrlBack provides the method to select the VRML file to be displayed. Click the *url* field to activate the Open button then click it or type the URL of a valid VRML file.

UrlFront, UrlRight, UrlLeft, UrlTop, UrlBottom enables you to specify a set of images that define a background panorama. The panorama consists of six images, each of which is mapped onto a face of an infinitely large cube, and centered in the VRML coordinate system. The images are applied individually to each face of the cube (*backUrl* to back face, *frontUrl* to front face, and so on).

Events:

Bind the background is visible or not visible in the scene when the incoming event is TRUE or FALSE.

GroundHoriz the incoming SFColor event specifies the ground color closest to the horizon. Repeats an incoming event as an outgoing one.

GroundNadir the incoming SFColor event specifies the Nadir (the bottom) ground color. Repeats an incoming event as an outgoing one.

SkyHoriz the incoming SFColor event specifies the sky color closest to horizon. Repeats an incoming event as an outgoing one.

SkyZenith the incoming SFColor event specifies the Zenith sky color. Repeats an incoming event as an outgoing one.

Triggered Button

Generates an outgoing event when a user clicks the button.

Properties:

color provides the method to choose the color from the palette of the Triggered Button object in the scene.

enabled indicates whether the Triggered Button is activated or not in the scene when you load the scene with the VRML browser. If FALSE, the button will not generate outgoing events.

transparency defines the degree of transparency of the Triggered button object in the range from 0% (a completely opaque surface) to 100% (a clear surface).

Events:

click generates a SFTime event when a user clicks the button in the Preview mode.

color the incoming SFColor event specifies the color of the Triggered Button object in the scene. Repeats an incoming event as an outgoing one.

enabled the Triggered Button is/is not activated in the scene when the incoming event is TRUE or FALSE. Repeats an incoming event as an outgoing one.

off generates an SFTime event to turn something off. The event is generated only for the odd numbered user clicks, i.e. the second, the fourth and so on.

on generates an SFTime event to turn something on. The event is generated only for the even numbered user clicks, i.e. the first, the third and so on.

toggle generates a Boolean value that is either TRUE or FALSE.

transparency the incoming SFFloat event specifies the degree of transparency of the Triggered button object (a sphere). Repeats an incoming event as an outgoing one.

DirLight

A directional light is a light that illuminates a scene with parallel rays. It affects only sub-objects, that is, objects that are children of the DirLight object.

Properties:

ambientIntensity affects the ambient (indirect) lighting for lit objects. The ambient lighting is computed (for red, green and blue) by multiplying the light's intensity by its ambientIntensity and multiplying the result for that color component. The ambient intensity may range from 0.0 (no light emission) to 1.0 (full intensity).

color specifies the spectral color properties of both the direct and ambient light emission as an RGB value.

direction specifies the direction vector of the illumination.

Intensity specifies the brightness in the range from 0.0 (no light emission) to 1.0 (full intensity).

on indicates whether the light is turned On (TRUE) or Off (FALSE).

Events:

Events correspond to the object's properties description. All incoming events are repeated as outgoing ones.

Fog

Defines a region of decreased visibility, to simulate fog, smoke, or smog.

Properties:

Bind allows you to activate the current object in the *ISA* editing mode and therefore to reflect changes from it in the scene. When several Fog objects are in the scene, only one object may be active at a time. In the editing mode, *ISA* binds the first Fog object encountered in the Scene Tree window. To activate an object, click anywhere in the *Bind* field of the Properties/Events and then the button that appears.

color the color of the fog.

type the rate at which fog thickens as distance from the viewer increases. Types are LINEAR, EXPONENTIAL.

visibilityRange the maximum distance at which a user can see anything through the fog, in metres; a value of 0 or less means no fog.

Events:

color the incoming event of the SFFloat type specifies the color of the fog. Repeats an incoming event as an outgoing one.

isBound indicates when the Fog becomes bound (TRUE) or unbound (FALSE).

set_bind if TRUE, makes this Fog the current one.

visibilityRange sets the maximum distance at which a user can see anything through the fog.

Group

The auxiliary object without any geometry that allows you to create additional levels of hierarchy in a scene. Once you group objects, you can treat them as a single object in your scene. For example, to group several objects for further animating, add the Group object to your scene, then move all needed objects in the scene hierarchy so that they become descendants of the Group object.

This is a nonrenderable object and has no any properties and events.

Inline

This object allows you to read a ready VRML model from anywhere on the Web or from your local hard drive and display it in the scene.

Property:

url provides the method to select the VRML file to be displayed. Click the *url* field to activate the Open button, then click it or type the URL of a valid VRML file.

Event:

Event corresponds to the object's property description. The incoming event is repeated as an outgoing one.

Link

The Link object allows you to add an Internet shortcut to an object or picture in a scene. You can load a Web page or scene by sending the SFTime event to the Link object in the Routing Diagram window.

Note: To check how the object reacts, publish your scene and explore it using a VRML browser.

Properties:

Address provides the method to specify links or references to another file or scene. Click the Address field and enter the desired address for a Web page or scene.

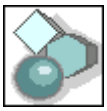
Params provides the method to enter additional information for the VRML or HTML browser. This information is a string in the form "keyword=value". For example, some browsers allow the specification of a 'target' for a link to display a link in another part of an HTML document. The parameter field is then: target=name_of_frame

Event:

load loads a page defined in the Address property field when an SFTime event is incoming.

a page defined in the Address property field when an SFTime event is incoming.

LOD



The LOD (level of detail) is used to allow browsers to switch between various representations of objects automatically. The level of object displayed is based on the distance between the object and the viewer. This distance is calculated from the viewpoint to the center of the geometry object. If the distance is less then the value specified in *switchDistance*, then the first level (defined by *urlHigh*) of the LOD is drawn. If the distance is greater then *switchDistance*, the second level is drawn.

Properties:

switchDistance defines the distance at which the level changes.

urlHigh provides the method to define a URL to an object with the highest detail.

urlLow provides the method to define a URL to an object with the lowest detail.

Events:

Events correspond to the object's properties description. All incoming events are repeated as outgoing ones.

Navigation Info

The Navigation Info object allows you to detail the values used to describe the virtual presence of the viewer within the world and the viewing model.

Properties:

avatarSize specifies parameters to be used in determining the viewpoint dimensions for the purpose of collision detection and terrain following. The first parameter specifies the allowable distance between the camera's position and any collision geometry in a VRML world before a collision is detected. The second value is the height above the terrain the viewpoint (camera) should be maintained. The third value is the height of the tallest object over which the viewer can move. All parameters are in metres.

Bind allows you to activate the current object in *ISA* editing mode and therefore to reflect changes from it in the scene. When several Navigation Info objects are in the scene, only one object may be active at a time. In the editing mode, *ISA* binds the first Navigation Info object encountered in the Scene Tree window. To activate an object, click anywhere in the *Bind* field of the Properties/Events and then the button that appears.

headlight specifies whether a VRML browser should turn on a headlight.

speed sets the rate at which the viewer travels through a scene, in metres per second.

type specifies a navigation paradigm. Types are WALK, EXAMINE, FLY, and NONE.

visibilityLimit sets the farthest distance the user is able to see.

Events:

avatarSize specifies parameters to be used in determining the viewpoint dimensions for the purpose of collision detection and terrain following.

headlight turns On or Off a headlight when the incoming event is TRUE or FALSE. Repeats an incoming event as an outgoing one.

set_bind sets parameters defined in the properties fields when the incoming event is TRUE.

IsBound repeats the *set_bind* incoming event as an outgoing one.

speed sets the rate at which the viewer travels through a scene, in metres per second.

type specifies a navigation paradigm.

visibilityLimit sets the farthest distance the user is able to see.

PointLight

A point light which can be located at a specific point in a scene and illuminates in all directions.

Properties:

ambientIntensity affects the ambient (indirect) lighting for lit objects. The ambient lighting is computed (for each element of red, green and blue) by multiplying the light's intensity by its ambientIntensity, and multiplying the result for that color component. The ambient intensity may range from 0.0 (no light emission) to 1.0 (full intensity).

attenuation specifies three coefficients to be used in calculating an attenuating factor. The first number is for the constant attenuation, the second for linear attenuation over distance, and the third for quadratic attenuation.

color specifies the spectral color properties of both the direct and ambient light emission as an RGB value.

intensity specifies the brightness in the range from 0.0 (no light emission) to 1.0 (full intensity).

on indicates whether the light is turned On (TRUE) or Off (FALSE).

radius indicates how far away from the light an object can be still lit.

sphereRadius defines the radius of the sphere that represents the PointLight object in the Perspective View in the scene editing mode, in metres.

Events:

Events correspond to the object's properties description. All incoming events are repeated as outgoing ones.

Proximity Sensor

Generates events whenever the user enters or leaves a defined box-shaped region.

Properties:

enabled indicates whether the sensor is turned on or off. If FALSE, the sensor is turned off.

Events:

enabled the ProximitySensor is/is not activated in the scene when the incoming event is TRUE or FALSE.

enter generates the outgoing event of the SFTime type whenever the user (a camera) enters a defined box-shaped region.

leave generates the outgoing event of the SFTime type whenever the user (a camera) leaves a defined box-shaped region.

toggle generates the Boolean outgoing event whenever the user enters (TRUE) or leaves (FALSE) a defined box-shaped region.

Sound

Provides a location and information on where a sound can be heard. The Sound object is located at a specific place in the scene, in the coordinates origin of the object (or the scene if Owner is None), which is the parent to the Sound. The direction of the sound is parallel to the (1 0 0) vector in the local coordinates. The region in which the sound can be heard is defined by two spherical regions. The inner one defines the region where the sound is played at the given intensity. In the region between the inner and outer sphere, the volume fades as a function of distance from the inner sphere.

Properties:

intensity adjusts the volume of the sound source; an intensity of 0 is silence, and 1 is the full volume.

Loop if TRUE, the sound is repeated indefinitely.

maxR defines the spherical region where the sound can be heard.

minR defines the spherical region where the sound is at full volume.

Speed lets you specify the factor to play a sound at different rates; a speed of 1 is normal playing.

sphereRadius defines the radius of the sphere that represents the Sound object in the Perspective View in the scene editing mode, in metres.

URL provides the method to select the sound file to be played.

Events:

Intensity adjusts the volume of the sound source when the incoming event is of the SFFloat type.

on if TRUE or FALSE, to play or stop playing a sound.

seton to play a sound when the incoming event is of the SFTime type.

SpotLight

The SpotLight object adds a light source that emits light from a specific point along a specific direction and constrained within a solid angle. A SpotLight object defines two cones of light: one with vertex angle equal to *beamWidth*, inside of which the light is at maximum intensity, and another coaxial light-cone with vertex angle equal to *cutOffAngle*. Light intensity drops off from the surface of the inner cone to the surface of the outer cone where it reaches zero.

Properties:

ambientIntensity defines the degree to which the light contributes to the world's ambient lighting. Ambient light in VRML simulates light that does not come directly from a light source. It is the general light that illuminates the entire scene. Ambient light is omni-directional and depends only on the number of light sources, not their positions with respect to the surface. The ambient intensity may range from 0.0 (no light emission) to 1.0 (full intensity).

attenuation defines the degree of attenuation to use. The first number is for constant attenuation, the second for linear attenuation over a distance, the third for quadratic attenuation (based on the square of the distance).

beamWidth defines the angle of the main cone of light, in degrees.

color specifies the spectral color properties of both the direct and ambient light emission as an RGB value.

cutOffAngle the angle beyond which nothing is lit, in degrees.

intensity specifies the brightness in the range from 0.0 (no light emission) to 1.0 (full intensity).

on indicates whether the light is turned On (TRUE) or Off (FALSE).

radius defines how far away from the light source an object can be lit. This assumes that the object is within the light cone.

SLConeHeight the height of the yellow cone that represents the SpotLight object in the Perspective View in the scene editing mode, in metres.

SLConeRadius radius of the cone that represents the SpotLight object in the Perspective View in the scene editing mode, in metres.

Note:

- It is strongly recommended that you change the size of the cone, representing the SpotoLight object in the editing mode, by specifying *SLConeHeight* and *SLConeRadius* (but not using the manipulators in the Perspective View).
- The object emits light along the cone axis from its vertex.

Events:

Events correspond to the object's properties description. All incoming events are repeated as outgoing ones.

Timer

The Timer object generates the outgoing Boolean events TRUE or FALSE and the SFTime type (tick) with the period defined by *cycleinterval.*

Properties:

CycleInterval the length of each cycle, in seconds. Must be greater than zero.

Enabled starts the timer whenever you load the scene with a VRML browser.

Loop if TRUE, the timer generates outgoing events indefinitely.

ToggleState defines the first Boolean value of the *toggle* outgoing event.

Events:

enabled if TRUE or FALSE, to start or stop a timer in a scene.

start starts the timer.

stop stops the timer.

tick generates the outgoing event (or repeated events if *Loop* is set to TRUE) of the SFTime type with the *Cycleinterval* delay when the timer starts.

toggle generates the outgoing TRUE event with the *Cycleinterval* delay when the timer starts. If the *Loop* is set to TRUE, it generates the events indefinitely: TRUE, FALSE, TRUE…

Viewpoint

Viewpoints are useful in those cases where you want the browser to control how the user views the scene. They describe a position and orientation for viewing the scene. You can also animate a viewpoint using the Animator to create a guided tour. Use this feature to define the exact path the user follows through the scene.

Properties:

AutoStore if TRUE, automatically stores the current camera position. This is useful in cases when you record an animation sequence for the guided tour. If FALSE, use *Store* to store a camera position.

Description a viewpoint name which the browser will show. Note that a descriptive name for the viewpoint will only appear in the list of viewpoints in Preview. Use Go To Viewpoint command from the pop-up menu of the Perspective View window to activate the viewpoint in the editing mode.

FieldOfView specifies a field of view from this viewpoint, in degrees. Note that the change of this field does not affect the field of view immediately. To examine changes, make sure the Viewpoint is selected, click the *GoTo* field at the Properties tab of the Properties/Events window and then the button that appears.

GoTo allows you to move the camera to the location of the viewpoint. Click anywhere in the *GoTo* field and then button that appears.

Jump controls a jump method from the old viewpoint to the new viewpoint.
TRUE – instantaneous transition to the new viewpoint.
FALSE – a jump with the transition effect.

Store allows you to store the current camera position as a viewpoint. To add a viewpoint to a scene, move the camera in the Perspective View or Plan to the desired location, click anywhere in the Store field, and then click the button that appears.

Events:

FieldOfView specifies a field of view from this viewpoint, in degrees. Repeats an incoming SFFloat event as an outgoing one.

bind the incoming Boolean event allows you to make the viewpoint active (TRUE) or inactive (FALSE) by routing it with any outgoing event.

bindTime generates the outgoing SFTime event when the viewpoint is activated.

isBound generates the TRUE or FALSE Boolean outgoing events when the viewpoint becomes active or inactive.

Visibility Sensor

The VisibilitySensor object allows you to detect when the user can see an object. When the object is visible, it can activate a behavior or animation. When the object is not in view it can deactivate the behavior or animation to improve the performance.

Properties:

Enabled

indicates whether or not the sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

Events:

Enabled

indicates whether or not the sensor is activated when you load the scene with the VRML browser. If False, the sensor is turned off.

enter

generates the SFTime event whenever any portion of the object bounding box becomes visible from the viewer's position.

leave

generates the SFTime event each time a box-shaped region exits when any portion of the object bounding box goes out of view.

toggle

generates the outgoing Boolean event when the object's box-shaped region enters the view (TRUE) or leaves it (FALSE).

WorldInfo

The WorldInfo object allows you to include information about the VRML world. This is strictly for documentation purposes and has no effect on the visual appearance or behaviour of the world. The *title* field is intended to store the name or title of the world. Any other information about the world can be stored in the *info* field, such as author information, copyright, and usage instructions.

Properties:

info provides the method to specify any number of strings to be in the *info* field of the *WorldInfo* VRML node of the outgoing VRML file. The default strings are titled as the following: Author, Company, and Version. Click the plus sign to change the string and use the appeared buttons to add or delete the string from the list.

title provides the method to specify the name or title of the world. Click this field to activate and then type the name you want.

TEXT

GetString (the string generator)

The GetString object allows you to specify and store an almost unlimited number of text strings that can be accessed and forwarded to the Text object. The number of items and their values (the text strings) can be specified by setting the object's properties. Depending on the incoming event *get_n_string* and *get_strings*, the object generates whether the string corresponding to the number or all strings are received.

Properties:

Strings provides the method to specify any number of strings to be stored. Click the plus sign to display the list of the text strings at Properties tab. Click the text item to activate the Add and Delete buttons. You can type the word or phrase for the selected item. Use the UTF-8 encoding for national letters in the text field. You can use, for example, ParallelGraphics ASCII to UTF8 Converter.

Events:

get_n_string the incoming event of the SFInt32 type that initiates the string generation from *string* with the number that equals the event value.

get_strings the incoming event of the SFTime type that initiates the generation of all text strings specified in *Strings.*

string the outgoing event of the SFString type with the value corresponding to the string with the *get_n_string* number.

strings the outgoing event of the MFString type with the values that contain all the text strings from *Strings*.

toStrConv (the string converter)

The toStrConv object converts a given Boolean, SFFloat, and SFInt32 values to a text string. This allows you to dynamically change strings in the Text object.

Properties:

addString specifies the text string that should be added to the beginning of every outgoing string. The field for the value may be empty. You type the word or phrase in Latin characters. Use the UTF-8 encoding for national letters in the *addString* field. You can use, for example, ParallelGraphics ASCII to UTF8 Converter.

IncIntValue if TRUE, adds 1 to the *intValue* value received when converting data.

Events:

intValue the incoming event of the SFInt32 type that generates the *stringValue* combining two strings: addString and a string equivalent for *intValue.*

floatValue the incoming event of the SFFloat type (a floating point value) that generates the *stringValue* combining two strings: *addString* and a string equivalent for *floatValue.*

boolValue the incoming event of the SFBool that generates the *stringValue* combining two strings: *addString* and a string equivalent for *boolValue* ("TRUE" or "FALSE").

stringValue the outgoing event of the SFString type with the value combining two strings: *addString* and a string equivalent for one of the incoming events.

CheckBox

The CheckBox object creates a check box control. The control is a small rectangle (check box) that has the specified text displayed to the right of it. When the user selects or clears the control, the control generates an event at *state_changed* jack.

Properties:

family indicates whether the text should be in a serif font, a sans-serif font, or a typewriter-like fixed-width font. The browser chooses a specific font to be used in each of those categories.

initialState sets the initial state of a check box. If TRUE, sets the check box state to checked, otherwise – to unchecked.

language specifies the context of the language for the text string.

Material sets the material properties for the check box. Click the Open button to start the Material Editor that provides functions to assign material to individual shapes and to create and edit surface materials.

name specifies text that is displayed to the right of the control. This provides the method to specify any number of strings to be displayed. Click the plus sign to display the list of the text strings at the Properties tab. Click the text item to activate the Add and Delete buttons. You can type the word or a phrase for the selected item.

size the height of each line of horizontal text, or the width of each line of vertical text.

spacing determines the distance, in lines, between consecutive lines of text. A value of 1 means single-spaced text; a value of 2 means double-spaced text.

style specifies whether the text should be in a roman (PLAIN), bold, italic, or bold italic typeface.

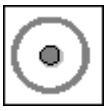
Events:

off generates SFTime event to turn something off. The event is generated when the user clears the check box.

on generates SFTime event to turn something on. The event is generated when the user marks the check box.

state_changed generates the Boolean value that is either TRUE or FALSE when the user change the state of the check box.

RadioButton



The RadioButton object creates a radio-button control. The control is a small circle that has the given text displayed to the right of it. The control highlights the circle and sends an event from the *state_change* jack when the user selects the button. The control removes the highlight and sends an event when the radio-button of the other control is selected. To create the group of radio-buttons, you have to always add at least two RadioButton objects and one RadioButtonControl object for every group of radio-buttons. You cannot use the only RadioButton object in a scene. To make the radio-button control work, you should define the RadioButton objects by adding names to the *Buttons* field of the RadioButtonControl object.

Properties:

family indicates whether the text should be in a serif font, a sans-serif font, or a typewriter-like fixed-width font. The browser chooses a specific font to be used in each of those categories.

initialState sets the initial state of a radio-button. If TRUE, sets the radio-button state to checked; otherwise – to unchecked. Note that changes are visible in the Preview mode or VRML browser.

language specifies the context of the language for the text string.

Material sets material properties for the radio-button. Click the Open button to start the Material Editor that provides functions to assign material to individual shapes and to create and edit surface materials.

name specifies text that is displayed to the right of the control. This provides the method to specify any number of strings to be displayed. Click the plus sign to display the list of the text strings at the Properties tab. Click the text item to activate the Add and Delete buttons. You can type the word or phrase for the selected item.

size the height of each line of horizontal text, or the width of each line of vertical text.

spacing determines the distance, in lines, between consecutive lines of text. A value of 1 means single-spaced text; a value of 2 means double-spaced text.

style specifies whether the text should be in a roman (PLAIN), bold, italic, or bold italic typeface.

Events:

off generates the SFTime event to turn something off. The event is generated when the user selects the other radio-button and therefore clear the current radio-button.

on generates the SFTime event to turn something on. The event is generated when the user select the radio-button.

state_changed generates Boolean value that is either TRUE or FALSE when the state of the radio-button is changed.

RadioButtonControl

Adding the auxiliary RadioButtonControl object makes the RadioButton objects capable of working in a scene. You should define the RadioButton objects by adding names to the *Buttons* field as they are shown in the scene.

Properties:

Buttons provides the method to choose the RadioButton in the scene. Specify the names of the RadioButton objects in order to make the group of radio-buttons enabled.

CurSelected sets the number of the initially selected radio-button. This numbering begins with zero and is specified with the *Buttons* property.

Event:

CurSelected the incoming event of the SFInt32 type that initiates the RadioButton with the number that equals the event value.

Sel_changed generates the SFTime signal when the selection is changed.

Text

The Text object allows you to add 2D text to a scene.

Properties:

color provides the method to choose the color of the Text object in the scene from the palette.

family indicates whether the text should be in a serif font, a sans-serif font, or a typewriter-like fixed-width font. The browser chooses a specific font to be used in each of those categories.

Horizontal indicates the direction of the text, whether the text reads horizontally (TRUE) or vertically (FALSE).

justify the first value indicates whether to line up the beginnings of the lines of text, to center each line of text, or to line up the ends. The second value indicates how to align the block of text in the minor direction.

size the height of each line of horizontal text or the width of each line of vertical text.

spacing determines the distance, in lines, between consecutive lines of text. A value of 1 means single-spaced text, a value of 2 means double-spaced text.

string provides the method to specify any number of string to be displayed. Click the plus sign to change the string and use the appeared buttons to add or delete the string from the list.

style indicates whether the text should be in a roman (PLAIN), bold, italic, or bold italic typeface.

transparency defines the degree of transparency of the Text object in the range from 0% (completely opaque surface) to 100% (clear surface).

Events:

color the incoming event of the SFColor type specifies the color of text in the scene. Repeats an incoming event as an outgoing one.

set_string the incoming SFString event that specifies the text string to draw.

set_strings the incoming MFString event that specifies multiple text strings to draw.

transparency the incoming event of the SFFloat type specifies the degree of transparency of text in the scene. Repeats an incoming event as an outgoing one.

Text3D

The Text3D object allows you to incorporate 3D text shapes into your VRML world and format it with any True Type font installed in your Windows system. Please note that this object uses Cortona specific VRML extension Text3D and therefore users should explore a scene containing these objects only with the Cortona VRML client. Other VRML browsers will not show 3D text properly.

Properties:

color provides the method to choose the color of the Text3D object in the scene from the palette.

depth specifies the thickness of each text string in the local coordinate system.

family indicates whether the text should be in a serif font, a sans-serif font, or a typewriter-like fixed-width font. The browser chooses a specific font to be used in each of those categories.

horizontal indicates the direction of the text, whether the text reads horizontally (TRUE) or vertically (FALSE).

justify the first value indicates whether to line up the beginnings of the lines of text, to center each line of text, or to line up the ends. The second value indicates how to align the block of text in the minor direction.

size the height of each line of horizontal text, or the width of each line of vertical text.

spacing determines the distance, in lines, between consecutive lines of text. A value of 1 means single-spaced text; a value of 2 means double-spaced text.

string provides the method to specify any number of string to be displayed. Click the plus sign to change the string and use the appeared buttons to add or delete the string from the list.

style indicates whether the text should be in a roman (PLAIN), bold, italic, or bold italic typeface.

transparency defines the degree of transparency of the Text object in the range from 0% (completely opaque surface) to 100% (clear surface).

Events:

color the incoming SFColor event that specifies the color of text in the scene. Repeats an incoming event as an outgoing one.

depth receives SFFloat event. Allows you to specify the thickness of each text string in the local coordinate system.

set_string the incoming SFString event that specifies the text string to draw.

set_strings the incoming MFString event that specifies multiple text strings to draw.

transparency the incoming SFFloat event that specifies the degree of transparency of text in the scene. Repeats an incoming event as an outgoing one.

TRANSFORM

MoveRepeater

Repeats or transforms an incoming SFVec3f *set_translation* event as an outgoing *translation_changed*. The transform is defined as translation along X, Y, and Z-axes multiplied by *move_scale.* The incoming events for positioning objects can be, for example, taken from the *Mover.translation_changed* eventOut which is added to the object by assigning Movable (Properties > Abilities list).

Create a route from *translation_changed* field to the *translation* field of the object you want to move. Note that to show the additional transformation fields on the object's icon in the Routing Diagram window, right-click outside the object's icon in the Routing Diagram, choose Advanced View.

When using events from MoveRepeater to change the object position, it is recommended that you create an additional level of transformation hierarchy with the Wrap by Group command for the object you want to move. This creates the new coordinate system where the object is not translated relative to the origin, and therefore it is possible to move the object smoothly by specifying its movement relative to the object's initial position.

Properties:

move_scale the factor that specifies the transformation with incoming *set_translation* event. If set to 1 (default value) – the outgoing event is repeated without transformation. Negative value means the movement is in the opposite direction. If greater than 1, the factor increases the translation.

Events:

translation_changed generates SFVec3f event to position any object in the scene whenever *set_translation* is received.

set_translation SFVec3f eventIn. The received components of the vector is multiplied by *move_scale* and then forwarded to *translation_changed.*

RotationInterpolator

The RotationInterpolator generates the SFRotation events to rotate objects smoothly in the scene about the selected axis. The rotation is defined as follows: *RevolutionCount* specifies the number of full turns in the direction according to the sign of the *Angle* property, while the *Angle* property adds the rotation between –360 and 360 degrees. If you want a rotation between –360 and 360 degrees, set the *RevolutionCount = 0.*

Create a route from the *rotation_changed* field to the *rotation* field of the object you want to rotate. Note that to show the additional transformation fields on the object's icon in the Routing Diagram window, right-click outside the object's icon in the Routing Diagram, choose Advanced View.

When using events from RotationRepeater to change the object rotation, it is recommended to create additional level of transformation hierarchy with the Wrap by Group command for the object you want to rotate. This creates the new coordinate system in which the object is not rotated relative to the coordinate axes, and therefore it is possible to rotate the object smoothly by specifying its rotation relative to the object's initial position.

Properties:

Angle specifies the rotation between –360 and 360 degrees and determines the direction of rotation..

Axis allows you to specify the rotation axis.

Loop if TRUE, the rotation is repeated indefinitely. If FALSE (the default value), a single cycle is performed.

RevolutionCount this specifies the number of revolutions (full turns).

TotalTime specifies the time of a single cycle.

Events:

finished generates the SFTime signal when the animation stops.

rotation_changed generates the SFRotation events that specify a rotation when the event at the *start* jack is received.

start the incoming event of the SFTime type that starts generating events for the rotation.

stop the incoming event of the SFTime type that stops generating events for the rotation.

RotationRepeater

Repeats or transforms an incoming SFRotation *set_rotation* event as an outgoing *rotation_changed*. The transform is defined as rotation about X, Y, or Z-axes multiplied by *angle_scale.* The incoming events for RotationRepeater can be, for example, taken from the *Rotator.rotation_changed* eventOut which is added to the object by assigning Turnable (Properties > Abilities list).

Create a route from *rotation_changed* field to the *rotation* field of the object you want to rotate. Note that to show the additional transformation fields on the object's icon in the Routing Diagram window, right-click outside the object's icon in the Routing Diagram, choose Advanced View.

When using events from RotationRepeater for the rotation, it is recommended to create additional level of transformation hierarchy with the Wrap by Group command for the object you want to rotate. This creates the new coordinate system in which the object is not rotated relative to the coordinate axes, and therefore it is possible to rotate the object smoothly by specifying its rotation relative to the object's initial position.

Properties:

angle_scale the factor that specifies the rotation with incoming *set_rotation* event. If set to 1 (the default value) – the outgoing event is repeated without transformation. Negative value means the rotation in the opposite direction. If greater than 1, the factor increases the degree of the rotation.

Axis specifies a direction for the rotation axis if the KeepAxis is set to FALSE.

KeepAxis allows you to specify a rotation axis. If TRUE (the default value) – the rotation axis is the same as for the incoming rotation. If FALSE, the rotation axis is defined by *Axis*.

Events:

rotation_changed generates the SFRotation event that specifies a rotation whenever the *set_rotation* is received.

set_rotation SFVec3f eventIn. The received components of the vector is multiplied by *angle_scale* and then forwarded to *rotation_changed.*

Screw

The Screw object allows you to simulate animations such as screwing the nut on the bolt. When rotated, such objects can be made to move into, or out of, the other object. The animation consists of two parts, the order of which you can select by specifying *ScrewFirst*: rotation and translation or only translation.

Create the following routes from the Screw object the object you want to animate: from *translation_chaned* to *translation* and from *rotation_changed* to *rotation.* Note that to show the additional transformation fields on the object's icon in the Routing Diagram window, right-click outside the object's icon in the Routing Diagram, choose Advanced View.

When using events from the Screw object, it is recommended that you create an additional level of transformation hierarchy with the Wrap by Group command for the object you want to animate. This creates the new coordinate system where the object is not initially rotated or moved.

Properties:

Axis allows you to specify the rotation axis.

LeftHanded enables you to define the direction of the rotation. If TRUE, it simulates a left-handed screw.

MoveLength allows you to specify the length to move the object without rotation.

MoveTime specifies the time of the translation cycle

ScrewFirst enables you to define whether two animations running simultaneously (translation and rotation) start playing first. If FALSE, the translation plays first.

ThreadLength allows you to specify the length of a screw thread.

ThreadsNumber specifies the number of revolutions (full turns).

ThreadTime specifies the time of the translation and rotation cycle.

Events:

rotation_changed generates the SFRotation events that specify a rotation when the event at the *start* jack is received.

translation_changed generates the SFRotation events that specify a rotation when the event at the *start* jack is received.

finished generates a signal when the rotation stops.

start the incoming event of the SFTime type that starts generating events for the rotation.

stop the outgoing event of the SFTime type is generated when the rotation is finished.